February 9th to June 19th, 2009
National Institute of Informatics (Tokyo)

Inductive Logic Programming Applied to Systems Biology

Master Degree Internship

Synnaeve Gabriel

Advisors:
Katsumi Inoue
& Taisuke Sato

Contents

Keywords:

inductive logic programming, molecular biology, systems biology, abduction, hypothesis finding, Michaelis-Menten kinetics, cell, bioengineering, inverse entailment, consequence finding, hidden Markov models, expectation-maximization algorithm, variational Bayes, binary decision diagrams, knowledge discovery process ...

Foreword:

This internship report is intended to an university jury but, more than that, the author considered to make it useful in another way: he would be very pleased if it could help someone (anybody) new in the research domain of "hypothesis finding for systems biology through logic-based methods". The content is intended to fullfill the absolutely basic requirements for entering the related research topics as well as giving the central references and showing the thought process that lead to such a conclusion.

1 Introduction

This internship is being done at the National Institute of Informatics (NII, Tokyo) under the supervising of Katsumi Inoue (NII) and Taisuke Sato (Tokyo Institute of Technology). The corresponding French advisor is Pierre Bessière (CNRS, Grenoble). The subject of the internship deals with the hypothesis finding (knowledge discovery) process for systems biology through inductive logic programming. The **goal** is to enhance the modeling of metabolic pathways of cells in order to obtain good predictive *in silico* models that will take all the mutual interactions of metabolites into account. The **interest** is to give a better understanding of the physiological state of the cell by improving the interpretation of the interactions between metabolic and signaling networks. The **application fields** range from biochemical engineering of proteins to the comprehension of cell aging (cancers) and drugs side effects prediction.

Systems biology is the discipline that studies biology at the molecular level by considering the cell as a complex system constituted of internal chemical interactions. The behavior of such a biological system is predicted from the point of view of a complex system involving time and non-linearity. The size and the non-linearity of the models of the cell's metabolisms forces us to predict some parameters that can't be measured. Some parameters can be measured *in vitro* but not *in vivo* and their values vary a lot because of all the external interaction of *in vivo* experiences. Numerical data is obtained at different levels: macroscopic, microscopic and molecular, and from different experiences. This leads us to set a threshold level between precision and generality by applying discretization.

Inductive logic programming is a machine learning technique that we use on experimental (sometimes incomplete) data for hypothesis finding. It allows us to compute either missing values (facts) or more general rules that explain some observations given a background knowledge. Finding missing facts is called abduction, and building rules (through generalization of clauses) is called induction. This techniques are very powerful to deal with the amount of experimental data and the increase of background knowledge in systems biology every year. Indeed, today's advances in data acquisition and handling technologies provides a wealth of new data that should lead to more predictive and comprehensive models. Besides, one can easily increment the background knowledge of a given model by adding computed rules or abducibles to it.

Logical kinetic models of the glycolysis and pentose phosphate pathways of *Escherichia Coli* and *Saccharomyces Cerevisiae* have been developped in order to study the metabolic response of a biological system after the injection of a pulse of glucose. The description of the internship's work follows this pattern: we will first explain the basic biological knowledge required to deal with metabolic pathways and explain the inductive logic programming framework. Then, we will draw a state of the art showing the current problems that scientists encounter. We will then explain our kinetic based approach and justify its use and the need for discrete levels. We finish by discussing the implementation and the results of logical abduction of the concentrations of metabolites before the dynamic transition. Before concluding, we will open on future works that are now possible.

2 Prerequisites and State of the Art

What makes the wealth and the complexity of a cell is on the one hand its numerous genes, and on the other hand the degree of interactions between its different chemical compounds (protein-protein, protein-ADN, protein-metabolite). It is therefore necessary to make an analysis taking into account the totality of interactions. For instance, if the databases concerning the *Escherichia Coli* increase each year with new results about the discovery of new proteins and new behavior due to some expressed genes, the mathematical modeling of the prokaryote cells is always an open question which excite the scientific community. Nowadays, bioinformatics represents the key field to explain the functionality of lifescience. To analyze a biological system it is necessary to find out new mathematical models allowing to explain the evolution of the system in a dynamic context [Kitano 02]. Many physical and biological phenomena may be represented on an analytical form using a dynamical system. The majority of kinetic models in biology are described by coupled differential equations and simulators are implemented with the appropriate methods to solve these systems. However, for most nonlinear dynamical systems it is difficult to find an analytical solution. The understanding of the phenomenon described by a complex system is carried out by a qualitative study of its behaviour such as stability or forking. The qualitative analysis based on perturbation method is a difficult task and is often performed by decomposition in subsystems which are simulated numerically.

2.1 Molecular Biology: the Cell

Basically, there are two types of cells but both have to provide functions and to reproduce. Both require a lot of intermediate steps that make use of **energy** and **material**.

2.1.1 Prokaryote and Eukaryote

There are many types of cells and they have many different functions, but we can extract some common metabolic pathways, when studying cells at a molecular scale. These metabolic pathways represent the chemical reactions and transformation of cell's components. Their detailled study would require an entire book, so we will only give points that were necessary to the comprehension of the presented problem. We have worked on models of *Escherichia Coli* and *Saccharomyces Cerevisiae* because they are representatives of two big families of cells: respectively, the prokaryote and the eukaryote. Prokaryote cells lack a nucleus (*karyon*) nor any other membrane-bound organelles (functional subunits). On the contrary, eukaryote cells have inner membranes, as seen in Fig.1, that bounds the nucleus and mitochondria for instance. Prokaryote cells are mainly constituting unicellular organism whereas eukaryote cells are usually 10 to 1000 times bigger (in volume) and found in multicellular organisms. Both types of cells can be seen as chemical factories filled with water and proteins delimited by the plasma membrane.

2.1.2 Energy

Inside the cell, the energy is carried by adenosine triphosphate (ATP: $C_{10}H_{16}N_5O_{13}P_3$), which stores avaible energy in chemical bonds with phosphoanhydride. The energy is released during the hydrolysis of ATP: when a high-energy phosphoanydride bond is broken, by the addition of a water molecule, to form adenosine diphosphate (ADP). This energy can be used for instance for muscle contraction or biosynthesis of proteins. Two common way to produce energy that will be stored in ATP are:

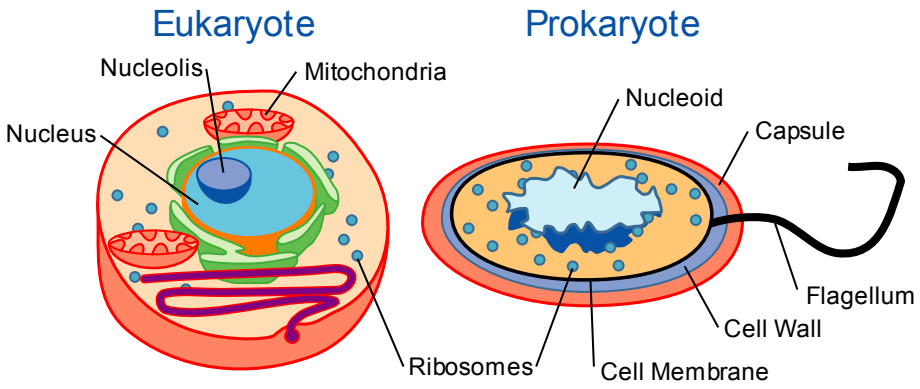


Figure 1: Cell types: left: eukaryote, right: prokaryote

- glycolysis: the degradation of glucose ($C_6H_{12}O_6$ for instance) in carbon dioxide (CO_2) and water (H_2O). The energy emitted by cutting chemical bonds is stored into ATP and reduced nicotinamide adenine dinucleotide (NADH).
- photosynthesis: the conversion of carbon dioxide and water into organic compounds (especially sugars) and oxygen (waste product) using solar energy. A part of the light energy absorbed by chlorophylls is stored in ATP.

2.1.3 Material

The universal material of the cell are proteins, which are complex molecules made from 20 basic amino-acids, 8 of them cannot be synthetised by human cells. Proteins form the internal skeleton and membrane of the cell. Other proteins can be (non-exhaustive list) sensors, intracellular messages, extracellular signals, enzymes (catalysing chemical reactions), motors, control gene activity (transcription factors) and carry substances across the plasma membrane. In a liver cell, protein accounts for approximately 20% of the weight with an approximative 7.0×10^9 total number of protein molecules per liver cell. The mean protein is 400 amino-acids long.

A big picture, *as in figure* 2, of the production of a given protein would be: a transcription factor reads the DNA and find the beginning of the template of the wanted protein. This template is copied into a single-stranded ribonucleic acid (RNA) wih the help of the RNA polymerase enzyme *(a)*. In eukaryotic cells, this RNA is shortened to form a "messenger RNA" (mRNA) *(b)* that will migrate through the cell to a ribosome *(c)*. The ribosome is a protein-RNA complexe that will use the mRNA as input and precise plan to assemble and link together amino acids to form the wanted polypeptide sequence (translation) *(d)*, symbol of the expression of the gene. The new polypeptide then folds into a functional three-dimensional protein molecule (post-translation) *(e)* and often even binds a biochemical functional group (acetate, phosphate, lipids, carbohydrates, etc.) *(f)*.

2.1.4 Metabolic Pathways

Molecular biologists search to model the cell by its biochemical interactions: protein-protein, protein-ADN, protein-metabolite. These reactions account for the function, the growth and the reproduction of the cell. For that purpose, they describe the chemical reactions occuring within the cell in a big graph connecting metabolites by annotated edges representing the reactions and their characteristics (enzymes, implied genes, constants,

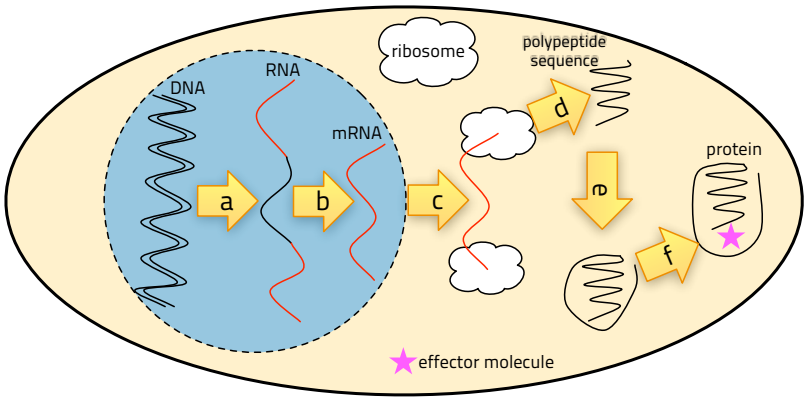


Figure 2: Protein synthesis originates in the nucleus (blue) and finish in the cytoplasm (beige). a: transcription ; b: post-transcription ; c: migration of the mRNA into the cytoplasm ; d: translation ; e: post-translation / folding ; f: binding of an effector

etc.): these interconnected reactions are forming a metabolic pathway. The substrates (reactants) of one reaction are the products of the previous one. All reactions are chemically reversible, but the thermodynamical conditions in the cell often favorise one direction. There are two types of metabolic pathways: catabolic ones, that break down input molecules to store energy, and anabolic ones, that construct big molecules from smaller units.

You can see an example of a metabolic pathway in appendix: Fig.24 from the Kyoto Encyclopedia of Genes and Genomes (KEGG) [Kanehisa & Goto 00, Kanehisa *et al.* 08]. This is the glycolysis pathway: glucose ($C_6H_{12}O_6$) enters the cell and is phosphorylated by ATP to glucose 6-phosphate (G6P) in order not to be able to leave the cell. The "goal" of the pathway is then to produce pyruvate ($C_3H_4O_3^-$) from this G6P and store the energy released when cutting bounds into ATP and NADH. This pathway can run in reverse to produce G6P for storage: in this case, it is called glucogenesis. Metabolic pathways are often regulated by inhibitions of some of the enzymes or by internal cycles (as the Krebs cycle, that is denoted as citrate cycle in Fig.24). Such metabolic pathways are specialised for each cell with some specific reactions and different protein syntheses.

Metabolic engineering tries to produce metabolites such as amino acids, vitamins, organic acids, etc. from biochemical synthesis. It is crucial to understand (to be able to control) the flux distribution, regulation phenomena and control properties of the interesting metabolism [Doncescu *et al.* 07]. As intracellular fluxes cannot be measured, they are computed by solving a set of linear equations consisting of the mass balances equations of the intracellular metabolites. $S[m \text{ metabolites} \times n \text{ reactions}]$ being the stoichiometric matrix of the metabolic network, $v = (v_1, v_2, \ldots, v_n)$ the unknown fluxes of the n reactions, $C = (C_1, C_2, \ldots, C_m)$ the concentration of the m metabolites and r the accumulation rate of metabolites, we have:

$$S.v = \frac{dC}{dt} = r \tag{2.1}$$

In metabolic engineering, the system is considered to be able to reach steady states: when the input flux is equal to the ouput one, so when $r = 0$. It is then possible to compute v. Therefore, determining the structure of a metabolic network and its steady states is the first step towards the biosynthesis of molecules of interest and it is needed to build a kinetic model for *in silico* analysis of the metabolism.

2.2 Inductive Logic Programming

This (sub)section is **mainly based** on the Machine Learning 2004 paper from Katsumi Inoue: Induction as Consequence Finding [Inoue 04]. This is only boiled down to the essentials to have an overview of how are constructed the hypotheses in the following parts.

2.2.1 Vocabulary, Notations & Definitions

A *clause* is a disjunction of literals: $C = \{A_1, \dots, A_m, \neg B_1, \dots, \neg B_n\}$, with atomic A_i, B_i, can also be noted $C = \{B_1 \wedge \dots \wedge B_n \supset A_1 \vee \dots \vee A_m\}$. The empty clause is noted $\square$. A *positive* (*negative*) *clause* is a clause whose disjuncts are all positive (negative) litterals. A negative clause can also be named an *integrity constraint*. A *Horn clause* is a *definite* (only one positive litteral) or negative clause. Ex: $C' = \{\neg B_1, \dots, \neg B_n, A\} = \{B_1 \wedge \dots \wedge B_n \Rightarrow A\}$ is Horn (otherwise, it's *non-Horn*). A *unit clause* is a clause of length 1, only 1 litteral. A *conjunctive normal form* (CNF) is a conjunction of clauses, and a *disjunctive normal form* (DNF) is a disjunction of conjunctions of literals. A *clausal theory* Σ is a finite set (CNF) of clauses. If a clausal theory contains only Horn clauses, it is a *Horn program*, otherwise, the clausal theory is said to be *full* (with non-Horn clauses). A clause C subsumes a clause D if and only if (iff) there is a substitution θ such that $C\theta \subseteq D$. C is then told to be more general than D. C *properly* subsumes D iff C subsumes D but D does not subsume C. $\mu\Sigma$ denotes the set of clauses in a clausal theory Σ that are not properly subsumed by any other clause in Σ (μ stands for "minimal subsumption"). For a clausal theory Σ, a *consequence* of Σ is a clause entailed by Σ. The set of all consequences of Σ is noted $Th(\Sigma)$. Completeness refers to the ability to prove any formula that is true. Consistency refers to the impossibility to prove both P and $\neg P$ at the same time. A *production field* is a pair $[\mathbf{L}, Cond]$, where $\mathbf{L}$ is a set of literals closed under instanciation, and $Cond$ is a condition to be satisfied. For instance $\mathcal{P} = [p(\_, \_, \_)^+, \text{ length} \leq 1 \text{ and term depth} \leq 1]$ could be used to set the abductive bias by asking for the hypotheses to follow that form, where $p(\_, \_, \_)^+$ is the set of all positive literal whose predicate symbol is p and takes 3 arguments, and satisfy this conditions. A production field $\mathcal{P}$ is stable if, for any two clauses C and D such that $C \subseteq D$, $D \in \mathcal{P}$ only if $C \in \mathcal{P}$.

2.2.2 Overview

Both induction and abduction are part of the inductive logic programming (ILP) framework [Mooney 97]. Their goals (see Fig.3) are to seek hypotheses that (2.2) account for given observations or examples while (2.3) staying consistent with the background. Given a background theory B and positive examples E (both are clausal theories), the task of induction and abduction is to find an hypothesis H such that:

$$B \wedge H \models E \tag{2.2}$$

$$\text{and } B \wedge H \not\models \bot \tag{2.3}$$

Abduction infers direct causes of observations, like missing facts (that hasn't been observed), that are often called *explanations*. **Induction** focuses more on finding general hypotheses that cover as many positive examples as possible (and as less negative ones as possible). Their relation and interactions are studied more in depth in [Flach & Kakas 00].

The resolution is complete for *consequence finding* [Lee 67] and even $C \models D$ can be replaced by $C \subseteq D$ (subsumes). With $\Sigma \models_{resolution} Th(\Sigma) \Rightarrow \Sigma \subseteq Th(\Sigma)$, $\Sigma \models \mu\Sigma$ and $\mu\Sigma \subseteq \Sigma$:

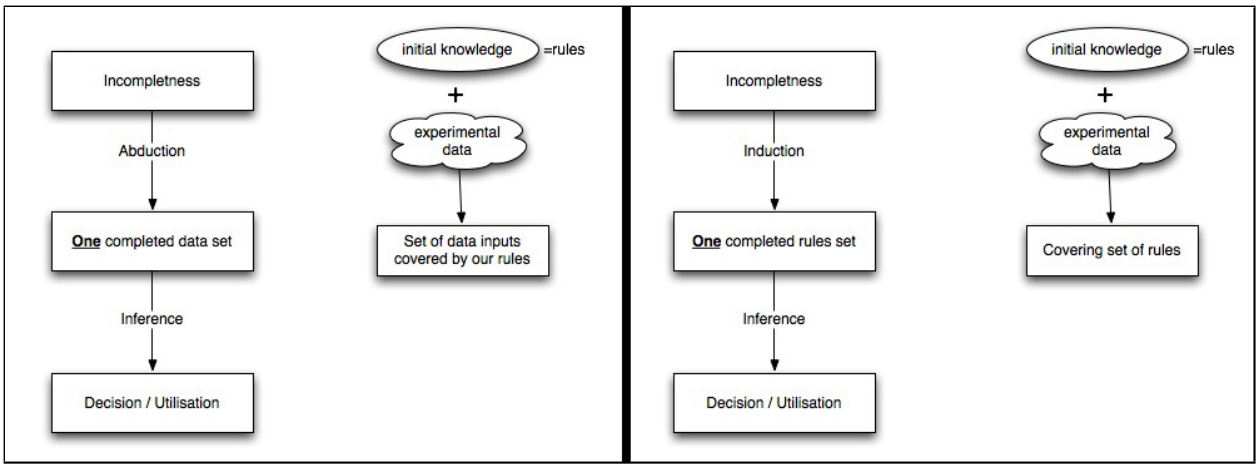


Figure 3: Left: abduction, Right: induction.

$Th(\Sigma)$ and $\mu Th(\Sigma)$ are equivalent under subsumption. Consequence finding a key point for ILP as it both provides a theoretical background for discussing the completeness of ILP systems [Nienhuys-Cheng & De Wolf 97] but also as it allows to use resolution (or tableaux calculus) for finding H.

Generalization is often required for induction in order to generate "general" hypotheses. Its task is, given a CNF F, to find a more general CNF H such that $H \models F$. Some well-known techniques for achieving that are:

- Reverse Skolemization: conversion of Skolem functions to existentially quantified variables.
- Anti-instanciation: ground terms are replaced with variables.
- Anti-subsumption (dropping of literals): some literals are arbitrarly dropped from a clause.

2.2.3 Inverse Entailment for Abduction and Induction

Abduction through IE is considered in [Inoue 92, Muggleton 95]. It uses the facts that:

$$(2.2 \ \& \ 2.3) \iff \quad B \wedge \neg E \models \neg H \tag{2.4}$$

$$and \quad B \not\models \neg H \tag{2.5}$$

Let $\mathcal{L}$ be the representation language to express our hypotheses, and Γ be a set of candidate hypotheses (so, for abduction, ground literals), defined as a subset of $\mathcal{L}$. We want to explain a finite number of observations $E_1, \ldots, E_n$ from an abductive theory (B, Γ). Γ is called the *abductive bias* as it restraint the possible expressions of the hypotheses. And so, with (2.4) and (2.5), the negation of H can be computed through a consequence finding procedure (or other entailment calculus) from $B \wedge \neg E$. We can consider:

B: a full clausal theory (containing non-Horn clauses)

E: a conjunction of existentially-quantified literals

H: a conjunction of literals (within the abductive bias)

With respect to this, the IE calculus is sound and complete for computing abductive explanations.

IE for **induction** was first considered in [Muggleton 95] with the Progol system. The principle that it introduced for induction as IE is to work with a "bridge" formula U such that

$$B \wedge \neg E \models U, \quad U \models \neg H \tag{2.6}$$

IE for induction as in [Muggleton 95] uses the bottom clause (conjunction of all unit clauses entailed by $B \wedge \neg E$) as "bridge" formula:

$$\bot(B, E) = \{\neg L | L \text{ is a literal and } B \wedge \neg E \models L\} \tag{2.7}$$

and a hypothesis H is constructed by generalizing a sub-clause of $\bot(B, E)$:

$$H \models \bot(B, E) \tag{2.8}$$

The negation of H can be computed through a consequence finding procedure. We can consider:

B: a Horn program

E: a Horn clause

H: a Horn clause

With respect to this, the IE calculus is sound (correct) but incomplete [Yamamoto 97] for computing inductive hypotheses.

2.2.4 CF-induction

By extending the notion of consequence finding, [Inoue 92] defined *characteristic clauses* to represent "interesting" clauses for a given problem. For a clausal theory Σ, the set of logical consequence of Σ belonging to the production field $\mathcal{P}$ is denoted as $Th_{\mathcal{P}}(\Sigma)$. The *characteristic clauses* of Σ with respect to $\mathcal{P}$ are defined as:

$$Carc(\Sigma, \mathcal{P}) = \mu Th_{\mathcal{P}})(\Sigma) \tag{2.9}$$

Which leads to $Carc(\Sigma, \mathcal{P}) = \{\square\} \Leftrightarrow \Sigma$ is unsatisfiable and $\mathcal{P}$ is stable. For instance, when $\Sigma = (\neg p \vee q) \wedge p$:

$$Carc(\Sigma, \mathcal{L}_\Sigma) = p \wedge q$$

There are several procedures to compute characteristic clauses (through the elegant $NewCarc$: new characteristic clauses expansion) that can be found in [Inoue 92] and [Inoue 04].

CF-induction (as induction in the large) is interested in the formulas derived from $B \wedge \neg E$ that are not derived from B alone. Instead of $\bot(B, E)$, it considers some clausal theory $CC(B, E)$ as a "bridge" formula. (2.4) and (2.5) can then be rewritten as:

$$B \wedge \neg E \models CC(B, E) \tag{2.10}$$

$$CC(B, E) \models \neg H \qquad (\Longleftrightarrow H \models \neg CC(B, E)) \tag{2.11}$$

$CC(B, E)$ is obtained by consequence finding from the characteristic clauses of $B \wedge \neg E$ because (by definition of $Carc$) any other consequence of $B \wedge \neg E$ belonging to $\mathcal{P}$ can be obtained by constructing a clause that is subsumed by a characteristic clause. Hence, we have:

$$Carc(B \wedge \neg E, \mathcal{P}) \models CC(B, E) \tag{2.12}$$

with $\mathcal{P}$ giving an *inductive bias*. In (2.11), we have $\neg CC(B, E)$ as DNF, it should be converted into the CNF formula F which H entails:

$$F \equiv \neg CC(B, E) \tag{2.13}$$

$$H \models F \tag{2.14}$$

Finishing the process of CF-induction from (2.13) is only to **generalize** (see page 8) the clausal theory F to H such that $B \wedge H$ is **consistent** and H is **Skolem free** (without Skolem terms, i.e. without constants that could be replaced by existentially quantified variables).

CF-induction can be tracted to a consequence finding procedure and consider the most general class:

B: a full clausal theory

E: a full clausal theory

H: a full clausal theory

The IE calculus is sound (correct) and complete for computing inductive hypotheses.

2.3 Previous and Related Works

"To understand biology at the system level, we must examine the structure and dynamics of cellular and organismal function, rather than the characteristics of isolated parts of a cell or organism." [Kitano 02]

2.3.1 Analytical Models and Quantitative Analysis

There are analytical models of cells' metabolisms, which are organised as complex networks of interconnected reactions, where the global behavior is the result of individual properties of enzymes, stoichiometry, metabolites, reactions constants. The approaches are based on mass-balance and/or kinetic methods which also account for dynamic behavior. The choice of Michaelis-Menten formalism has often been made as a representation of a non-linear allosteric regulation system.

Dynamic models are often based on integral and (ordinary) differential equations (ODEs). In [Waser *et al.* 83], the authors study the kinetics of the phosphofructokinase reaction (part of the glycolysis pathway) through the use of ODE. They used mainly Hill's kinetics for the regulation of the enzymatic activity. A work that is closer to what we have done can be found in [Franco & Canela 84], where the authors simulate the purine metabolism by ODEs (one equation per reaction-enzyme couple) using the Michaelis constant of each enzyme. The results that we have for the experiment of a pulse of glucose on *Saccharomyces Cerevisiae* are based on an simulator from Andrei Doncescu using integral / differential equations. This is a work based on experimental data and

[Mauch *et al.* 00] where the authors describe the experiments, assumptions and modeling in details. Not every quantitative analysis is done through ODE. For instance, the paper [Hofestädt & Thelen 98] presents a quantitative simulation of biochemical networks while using Petrinets.

In [Reder 88], the author has emphasized on the structural characterization and properties of the metabolic network as opposed to the reaction kinetics. This is not directly related to the current work, but this is an interesting approach of a very related problem. More generally, the analytical models are designed to have a very precise modeling of the cell (or part of it) but lack a global approach that would allow to find "general" formulas that rule this complex system. Also, it is not always possible to measure the evolution of concentration of metabolites *in vivo*, but parameters of ODEs (or other quantitative) models are sometimes approximated by *in vitro* measurements. This can be a source of error as they can differ a lot from *in vivo* numbers. This is why logic based approaches (and mixed ones) exist: it should be better at handling general cases and structural driven rules.

2.3.2 Logic Based Approaches

In [Tamaddoni-Nezhad *et al.* 06], the authors use both abduction and induction to model inhibition in metabolic pathways. Abduction is used to complete the background knowledge, and then induction allows for learning general rules while working in an hypothesis language that is disjoint from the observation language. They clearly aim at predicting the inhibitory effects of different substances to assess the potential harmful side-effects of drugs. The network topology and the functional classes of inhibitors and enzymes constitute the background knowledge, whereas the examples are derived from *in vive* experiments involving nuclear magnetic reasonance analysis of metabolite concentrations in rat urine following injections of toxins. The background knowledge is regarded as incomplete and so the "hypotheses are considered which consist of a mixture of specific inhibitions of enzymes (ground acts) together with general (non-ground) rules which predict classes of enzymes likely to be inhibited by the toxin". Their results suggest that non-ground hypotheses have a better predictive accuracy than ground ones when sufficient training data is provided.

[Chen *et al.* 07] investigated probabilistic inductive logic programming (PILP) application to systems biology within the stochastic logic programs (SLP) description. Their example data was derived from studies of the effects of toxins on rats metabolic pathways reactions inhibitions (using nuclear magnetic resonance). They found a decrease in error compared to classic ILP but the main advance was to be able to learn PILP models from probabilistic examples. They used the same data to make a comparative study [Muggleton & Chen 08] of existing probabilistic logic models, being them statistical relational model (SRM) based as in [Kersting & De Raedt 01] or statistical logic programming (SLP) based as in [Sato 98].

A publication in Nature [King *et al.* 04] makes a strong point for the automation of the scientific discovery process. It is stated that the data are being generated much faster than they can be analysed by experts. Thus, the authors have built a physical robot scientist that conduct cycles of scientific experimentation through artificial intelligence techniques. A cycle is as follows: the system automatically generates hypotheses to explain

observations, come up with the testing experiments, physically runs them using a laboratory robot, interprets the results and change the discovered hypotheses in consequence, and cycle again. The authors of this paper applied it to the determination of gene function in the growth of *Saccharomyces Cerevisiae* (yeast). They show that this automated experiment stategy is competitive with human performance for experiment selections.

In [Ray *et al.* 09], the authors show how biological pahways can be modeled in a logic programming formalism. They then use a nonmonotonic logic system for learning and revising such networks from observations. There are numerous logic-based attempts to model cells metabolisms, we should also cite BIOCHAM from [Chabrier-Rivier *et al.* 04], which stands for BIOCHemical Abstract Machine and that consist in modeling the biochemical system through reaction rules and properties. The additional knowledge from experiments comes is formalized in temporal logic and can be used to learn modifications or refinements of the model. The biological validation can be done through model-checking, both qualitatively and quantitatively.

To conclude this overview, in [King *et al.* 05], the analysis of metabolic pathways is done by using a qualitative reasoning (QR) as a unified formalism for prediction (simulation) and identification. This QR formalism is an abstration of ordinary differential equations (ODEs) that has the advantage over logical graph-based models of explicitely including dynamics. These QR models are obtained from generate-and-test ILP procedure, which bring it close to our approach.

2.3.3 Japanese-French Symposium on Systems Biology

The goal of the Japanese-French symposium on systems biology is to elaborate symbolic models of these systems in order to discover the mechanisms that govern them, so as to better apprehend the behavior of cells. Previous work has been done concerning [Doncescu *et al.* 07, Yamamoto *et al.* 08] the use of inductive logic programming for studying metabolic pathways.

In [Doncescu *et al.* 07], the authors explain a practical use of CF-induction to explain the metabolism of a simple, yet relevant, pathway that has a hidden metabolite and a bidirectional reaction. Their framework is the one of metabolic flux analysis (2.1) with `up/down` variations of concentrations. In [Yamamoto *et al.* 08], the authors explain the rules that they use for toxins inhibitory effects within metabolic flux analysis. This rules are:

$$con(X, up) \leftarrow reac(Y, X) \wedge reac(X, Z) \wedge con(Y, up) \wedge \neg inh(Y, X) \wedge inh(X, Z) \quad (2.15)$$

$$con(X, down) \leftarrow react(Y, X) \wedge con(Y, down) \wedge \neg inh(Y, X) \quad (2.16)$$

where $con(metabolite, change)$, $reac(X, Y)$, $inh(X, Y)$ respectively stand for concentration(metabolite, change), reaction between metabolites X and Y, inhibition of the reaction between X and Y. We have to keep in mind that inhibition is indeed an effect of the toxin on the enzyme that normally allows for the reaction activation.

In [Inoue *et al.* 09], the work is based on the same models and problem than in [Tamaddoni-Nezhad *et al.* 06, Chen *et al.* 07, Muggleton & Chen 08]: the inhibitory effects of toxins on the rat metabolisms. What is new in this work, and that we have

successfully used in the current one, is that it describes a method to rank hypotheses with the help of BDD-EM [Ishihata *et al.* 08]. That is also in this paper that is mentioned for the first time the full system for hypothesis finding, described in Fig.4, that we will base our work on.

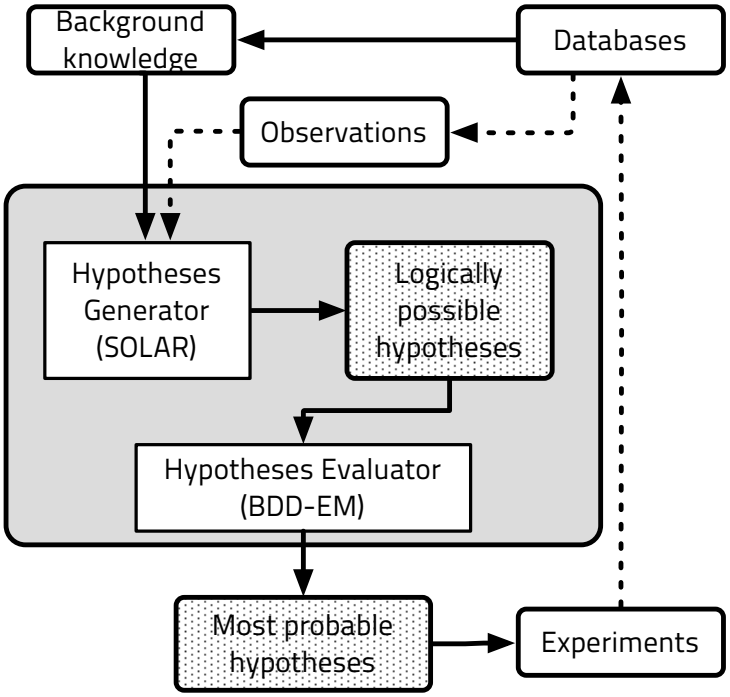


Figure 4: Current biological hypothesis-finding system as explained in [Inoue *et al.* 09]

3 Discrete Levels and Kinetic Modeling of Reactions

The kinetic model that will be presented in the following allows for the study of metabolic flux analysis by handling not only qualitative results (like changes of concentration of a metabolite between to instants) but quantitative values. For that, we clusterize continuous concentrations of metabolites over time into discrete levels and discrete timesteps. Then (4.3), we applied it on an inverse problem: given the measured concentrations of some metabolites in steady state, we compute the concentrations of metabolites before the dynamic transition from the perturbation that cause a pulse of glucose to this steady state thanks to the kinetic modeling.

3.1 Precision – Generality trade-off

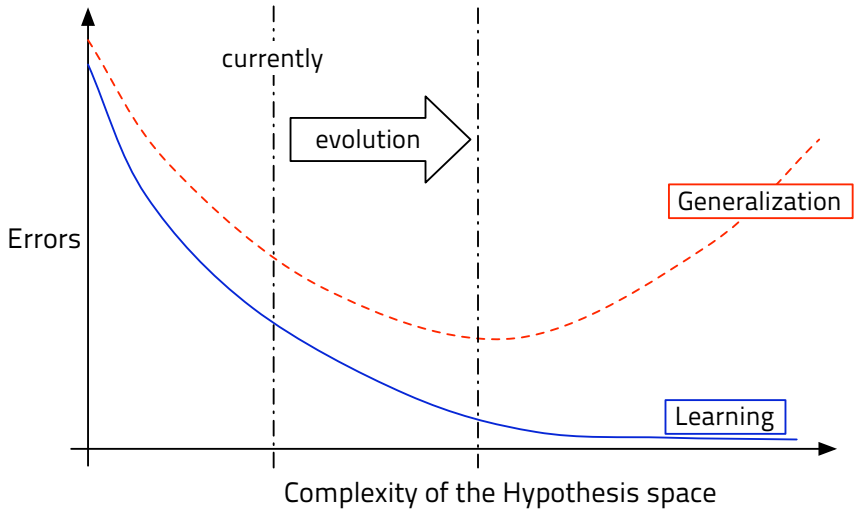


Figure 5: Variations of the learning and generalization errors in function of the complexity of the hypothesis space. The "evolution" arrow show where we are heading by discretizing the time series more finely.

We are dealing with the classic problem of machine learning: given some samples data as input, could we learn hypotheses that describe this data in terms of important properties, for us to be able to predict the class (or what will happen) when we are given a new sample (an event occurs). This is also the way we, as human being, learn our lessons, make hypotheses, conclusions, and even learn to walk. Let $\mathcal{X}$ be the samples space and $\mathcal{H}$ be the hypotheses space. These spaces are generated respectively by $\mathrm{L}_{\mathcal{X}}$, language of the samples, and by $\mathrm{L}_{\mathcal{H}}$, language of the hypotheses. The task of machine learning is to find a function $\mathcal{X} \to \mathcal{H}$ that associate the $x \in \mathcal{X}$ with corresponding classes $h_i \in \mathcal{H}$. The problem is that the x can be differing from each others by very tiny differences in their features. Which features are the most important for $\mathrm{L}_{\mathcal{H}}$? If we try and make a machine that learns by taking all the features into account, that is equivalent to have $\mathrm{L}_{\mathcal{X}} = \mathrm{L}_{\mathcal{H}}$ and learning is simply copying the values of the features. When a new x will be given to this machine, it will be unable to differentiate that sample from the others, in fact: every sample will be a special case.

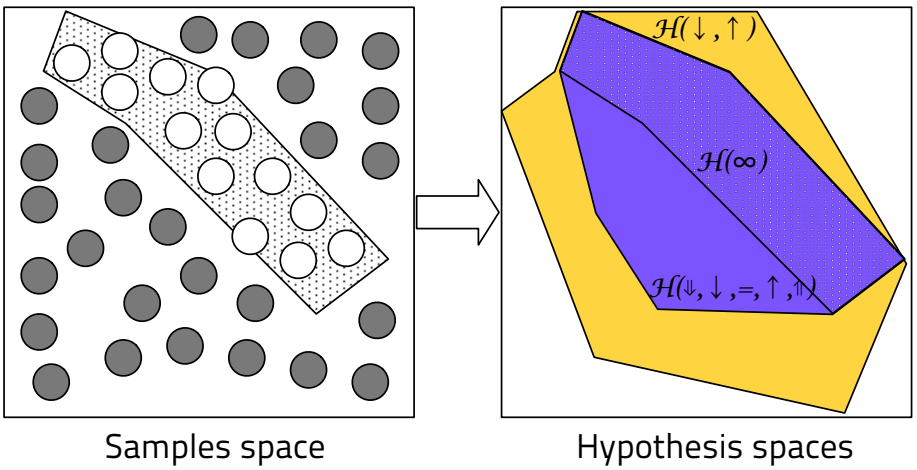


Figure 6: Trade-off between the ability to describe the samples space given the complexity of the hypothesis space

There is a recurrent problem in machine learning: the learning error is decreasing when the dimension of the hypotheses space is growing but the generalization error begins growing after some point as seen in Fig.5. As stated above, if the complexity of the hypotheses space is too big, the hypotheses will all be special cases. There are numerous of possible metaphors for explaining machine learning, one of them that is near the reality is that machine learning is like a data compression task: if the dictionary of possible features ($L_{\mathcal{H}}$) is too small, there will be a lot of loss in the process and the learning error will be big. If you specialise it too much, you will lose robustness (predictive power) and each compressed data input will be so precise that it will look like neither of the others. That's the strong argument in favor or Occam's razor: "entities should not be multiplied unnecessarily", meaning that if one has to choose between two hypotheses, the best choice should be the shorter (in terms of use of $L_{\mathcal{H}}$) one.

In regard to the previous results, it seemed that the limitations are due to a too small $L_{\mathcal{H}}$ with only `up` and `down` for the concentrations of metabolites. It doesn't allow us to have enough discrimination power between the samples (hypotheses are too simple) as showed in Fig.6. We hope to have a better (more accurate, yet still robust) predictive power by increasing the expressivity of the hypotheses (Fig.6). For example, if we work as in [Tamaddoni-Nezhad *et al.* 06, Inoue *et al.* 09], trying to analyze the inhibitory effect of toxins on metabolisms based on NMR measurements on rats: `up / down` correspond here to the differences between an experiment without toxins and one with an injection of toxins. If we have a piece of pathway as in Fig.7 and use the same rules as in [Yamamoto *et al.* 08], we are unable to infer if Reaction 1 (R1) is inhibited or not with just `up / down`. Now if we had a finer discretization of the values and that we found that Metabolite 3 (M3) has a big increase of concentration between the two experiments, but that M2 has only a low decrease: we can infer that the consumption (transformation) of M2 alone is not enough to account for M3 increase, and thus M1 should also be consumed and R1 is not inhibited.

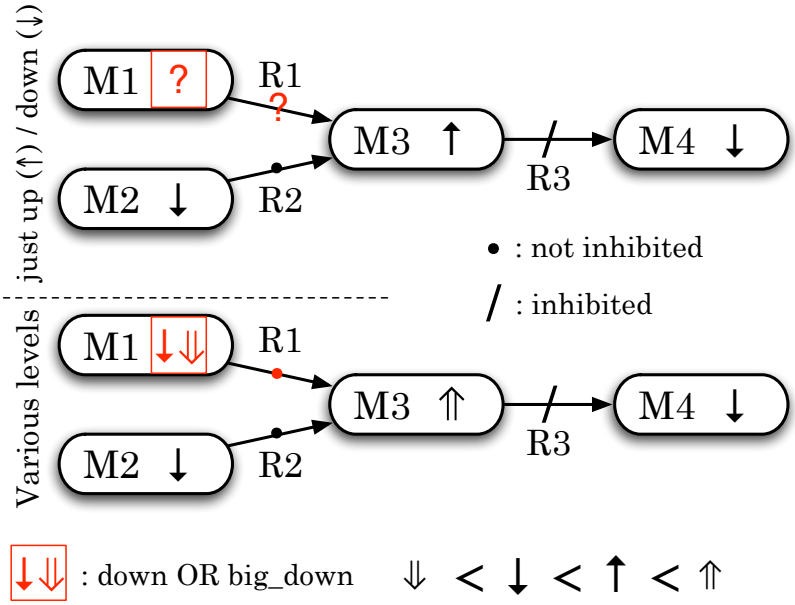


Figure 7: Being more expressive (4 levels instead of 2 as previously) allows for more deductions: we can infer that R1 *has to* be not inhibited and M1 *has to* be consumed.

3.2 Introducing Discrete Levels (in the Symbolic Modeling)

As seen before in Fig.7, one way to increment de complexity of the hypothesis space is by being able to deal with other qualifications than just "up" and "down". We have to be careful not to have a too fine discretization in order to avoid the right part of Fig.5: we would fail in our attempt to learn general hypotheses (and we would be doing numerical analysis, which is the area of expertize of analytical models, not ILP). That's the reason why we will use scores that penalize solutions for discretization with too much levels.

3.2.1 Time Series Discretization

In our modeling, we first introduce discrete concentration levels to filter what are the relevant changes of concentration of the metabolites, in regard to hypotheses generation from ILP. We need to be able to infer hypotheses that have a certain level of generality and, for that, we should use intervals instead of single real values. This could have been done with an interval constraints approach [Benhamou 94], but we currently choose a discretization approach. Although this gives us less freedom in the logic part as levels are fixed (as if we have had fixed intervals), levels can be handled just as symbols in a logical model of pathways, without having any knowledge of the real values behind it.

Discretizing time series is a research field in which many works have been conducted recently on the rather new problem of *unsupervised* discretization of time series. In the following, time is considered to be on the x-axis and value(s) on the y-axis. [Geurts 01] uses regression trees and tries to minimize the (y-axis) variance of the values in each time sample (x-axis interval). The drawback is that one can't directly control the time

sampling (nor the values, but this is closer to what we want with an unsuppervised clustering). [Keogh *et al.* 05] have developed "Symbolic Aggregate approXimation" (SAX) that is built on piecewise aggregate approximation and makes a normality assumption on measured values. In [Lin *et al.* 07], the authors showed the theoretical validity of SAX and we considered using it. However, we didn't see an easy way of dealing with our various and distant (in y-axis mean) time series because of the normality assumption: we should have one Gaussian for each time-series. [Mörchen *et al.* 05] proposed the Persist algorithm, that is really noise-resistant. But more of that, we have been interested in their exhaustive comparison of the quality of discretized sequences resulting from Persist, SAX and continuous hidden Markov models (CHMMs).

3.2.2 Our approach

In the end, both the facts that we had little to no noise[1] in our experimental time-series and that we wanted to discretize our concentrations time-series all at once (with the same levels) lead us to try and develop our own discretization method, adapted to (and taking benefit of) the particularities of our data. Our practical problem is that we want to have a statistically relevant (unsupervised) discretization for N metabolites concentrations over time. We also have to discretize the values of K_m (Michaelis-Menten constants), for each reaction, with the same levels.

For that purpose, we have chosen to use a probabilistic model, used in speech recognition and time-series analysis: continuous hidden Markov model (HMMs) [Rabiner 89]. We can therefore compute an appropriate number of levels (that was 3 for the experiment with *Escherichia Coli* , 9 for the one with *Saccharomyces Cerevisiae*) in regard to a Bayesian score such as Bayesian Information Criterion (BIC) [Schwarz 78] or as the Cheeseman-Stutz score [Cheeseman & Stutz 95] or as the variational free energy. This process can be achieved through maximum likelihood estimation or maximum a posteriori estimation [Gauvain & Lee 94] or through a variational Bayesian method [Beal 03, Ji *et al.* 06], respectively. There are more details in the implementation part beginning page 21.

Whereas the results of CHMMs discretization are relevant and already usable. We have also begun working on another discretization method that heavily uses the fact that we have almost no noise. One can (and is obliged to) apply a filtering as Savitzky-Golay [Savitzky & Golay 64] filtering beforehand if the data is noisy (see Fig.21 page 43). The choice of Savitzky-Golay filtering has been made as it keeps the amplitude of the time series (the maximum values) and it is particularly adapted to experimental data from chemistry [Savitzky & Golay 64]. The changes of tendencies of the concentration time-series seem important to be able to infer the cause-consequences relations with the inducted logic rules. With respect to this assumption, the inflexion points (see Fig.8) of the time series are central in this discretization approach. We use them as the seeds of y-values clusters as well as sampling stops for the time-sampling. We should now implement statistical tests to merge (prune) some levels (y-axis) and sampling stops (x-axis).

[1] Indeed, Andrei Doncescu provided a simulator that enables us to get very clean signals. Even without that, the signal/noise ratio of experimental data would allow us to smooth is efficiently beforehand.

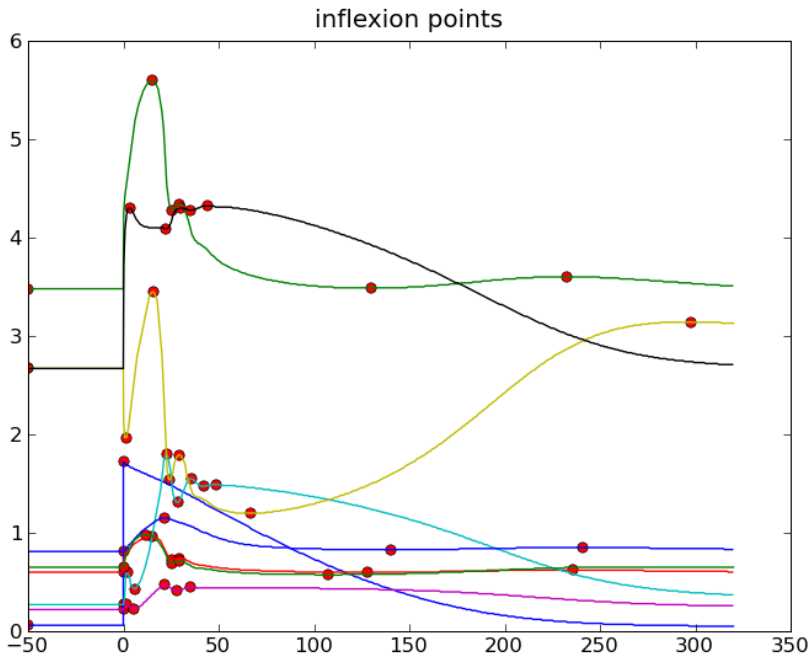


Figure 8: Inflexion points (red dots) of the evolution of the concentrations of metabolites in a "pulse of Glucose" experiment in *Saccharomyces Cerevisiae* .

3.3 Michaelis-Menten Kinetics

3.3.1 Establishing Michaelis-Menten Equation

The metabolic networks dynamics are in their enzymatic part ruled by the combination of classical kinetics: essentially Michaelis-Menten, Hill and allosteric ones. If we limit our modeling to these kinetics, we can highly simplify their mathematical handling. Michaelis-Menten treatment assumes that the two binding equilibria are fast when compared to the interconversion of ES and EP. The choice of Michaelis-Menten kinetic model have been made, because it is the more general representation for a non-linear allosteric regulation system. It assumes that the two binding equilibria are fast when compared to the interconversion of ES and EP.

Note: $[S]$ means "concentration of the metabolite S". SI unity is mol/L (you will also find $mmol/L$ for millimol/L). $1mol \approx 6.022 \times 10^{23}$ elementary entities, Avogadro constant.

$$E + S \rightleftharpoons_{k_{-1}}^{k_1} ES \rightarrow^{k_2} E + P \tag{3.1}$$

Under the quasi steady-state assumption, with to (2.1), it comes:

$$\frac{d[ES]}{dt} = k_1[E][S] - (k_{-1} + k_2)[ES] \approx 0 \tag{3.2}$$

$$\text{and } \frac{d[P]}{dt} = k_2[ES] \tag{3.3}$$

Under the assumption that the total quantity of the enzyme E is constant, it comes:

$$[E]_{total} = [E] + [ES] = const. \tag{3.4}$$

(3.4) into (3.2) gives:

$$k_1[S]([E]_{total} - [ES]) - (k_{-1} + k_2)[ES] = 0 \quad (3.5)$$

$$[S][E]_{total} = [S][ES] + [ES](\frac{k_{-1} + k_2}{k_1}) \quad (3.6)$$

$$\text{let } K_M = \frac{k_{-1} + k_2}{k_1} \quad (3.7)$$

$$\Longrightarrow [ES] = \frac{[S][E]_{total}}{K_M + [S]} \quad (3.8)$$

by replacing $[ES]$ in (3.3) and with the maximum velocity $V_m = k_2[E]_{total}$, it comes:

$$Michaelis - Menten\ equation : \frac{d[P]}{dt} = V_m \frac{[S]}{[S] + K_m} \quad (3.9)$$

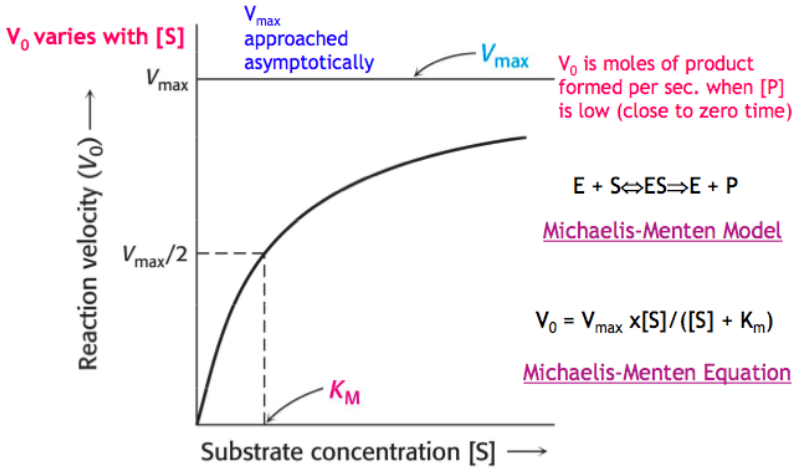


Figure 9: Reaction velocity given the substrate concentration with Michaelis-Menten kinetics (under quasi steady-state and constant enzyme concentration assumptions)

The Michaelis-Menten constants can be determined by a series of experiments with varying $[S]$ (substrate concentration) and for instance a Lineweaver–Burk plot, plotting the inverse of substrate concentration against the inverse of the initial velocity (see Fig.10).

3.3.2 Simplification

If both S and P are present, neither can saturate the enzyme. For any given concentration of S the fraction of S bound to the enzyme is reduced by increasing the concentration of P and *vice versa*. For any concentration of P, the fraction of P bound to the enzyme is reduced by increasing concentration of S. When we have $S \leftrightarrows P$, we just have to consider reactions for both directions. We consider a time discretization of the chemical rate equation for a reation between a Substrate and a Product with respective stoechiometric coefficient s and p:

$$s.S \rightarrow p.P \ : \ rate = \frac{1}{p} \times \frac{d[P]}{dt} \longrightarrow_{disc.time} \frac{1}{p} \times \frac{\Delta[P]}{\Delta T} \quad (3.10)$$

$$(3.9) \text{ and } (3.10) \Longrightarrow p \times rate = V_m \frac{[S]_T}{[S]_T + K_m} \approx \frac{[P]_{T+timestep} - [P]_T}{(T + timestep) - T} \quad (3.11)$$

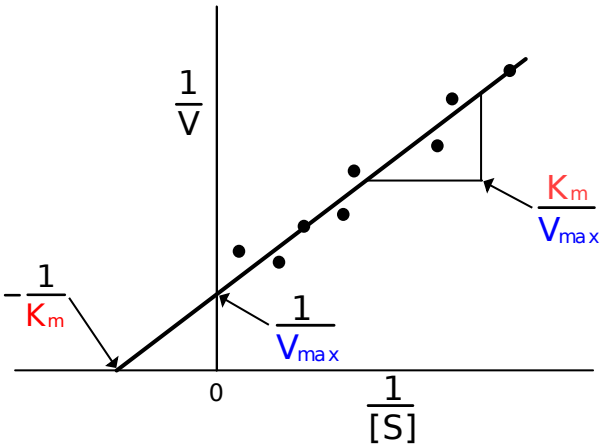


Figure 10: $(\frac{1}{[S]}, \frac{1}{V})$ plot, a.k.a. Lineweaver-Burk plot

If we choose to work with a **constant timestep**:

$$\implies [P]_{T+1} = V_m \frac{[S]_T}{[S]_T + K_m} + [P]_T \tag{3.12}$$

We can note that the Michaelis-Menten constants (`Km`) are homogenous to a concentration. We can then state `concentration(Km, Level, T)` ($\forall$ `T`) in our modeling to set them.

4 Implementations and Results

We have been developping an automated framework to deal with different real world pathways and experiments. It is currently composed of two tools:

- `kegg2symb`: this is a program that transforms metabolic pathways from the KEGG online database into symbolic relational models by querying KEGG's API (see Fig.23 page 44 in Appendix).
- The combination of an implementation of continuous HMMs [Gauvain & Lee 94, Ji *et al.* 06] with `py-tsdisc`, a Python automating wrapper that takes time-series of concentrations of metabolites and output the corresponding symbolic predicates `concentration(metabolite, level, time)`.

Both are under heavy developpement but already avaible for downloading and use through their working repository[2] under the free BSD/MIT-Python licenses.

Along with that, we used SOLAR [Nabeshima *et al.* 03] for the hypothesis finding part and BDD-EM [Ishihata *et al.* 08] for ranking these hypotheses. You can see the whole path of the data, from experimental results and KEGG database to the ranked hypotheses in figure 11 page 22. The "new" parts (this work) are the one with a grey background.

4.1 `kegg2symb`: Automatic Conversion of Pathways

This program generates a symbolic model with the name of a pathway from KEGG. `http://www.genome.jp/dbget-bin/show_pathway?sce00010` is described in `ftp://ftp.genome.jp/pub/kegg/xml/organisms/sce/sce00010.xml`.

The default behavior is to parse the XML describing the map and to output the reaction nodes. As all that contains the XML are IDs, it has the options:

- offline: builds the reactions with KEGG's {enzyme:id} and {metabolite:id} with one `reaction` predicate per enzyme.
- genes: reactions with genes that have to be activated to produce such a protein, and metabolites names (from KEGG's API). One `reaction` predicate per gene.
- convert (default): reactions with metabolites and enzymes names (from KEGG's API) with one `reaction` predicate per enzyme.

We think that this is interesting to automate such uses of existing databases to be able to build large models quickly and to be able to persue our automated knowledge discovery process.

4.2 Discrete Levels

4.2.1 Hidden Markov Models

An hidden Markov model (HMM) is a statistical model in which the system being modeled is assumed to be a Markov process with unobserved state: the corresponding discrete

[2] kegg2symb: `http://github.com/SnippyHolloW/kegg2symb/tree/master` from Gabriel Synnaeve
HUP: `http://sato-www.cs.titech.ac.jp/kameya/hup4d/` from Yoshitaka Kameya
py-tsdisc: `http://github.com/SnippyHolloW/py-tsdisc/tree/master` from Gabriel Synnaeve

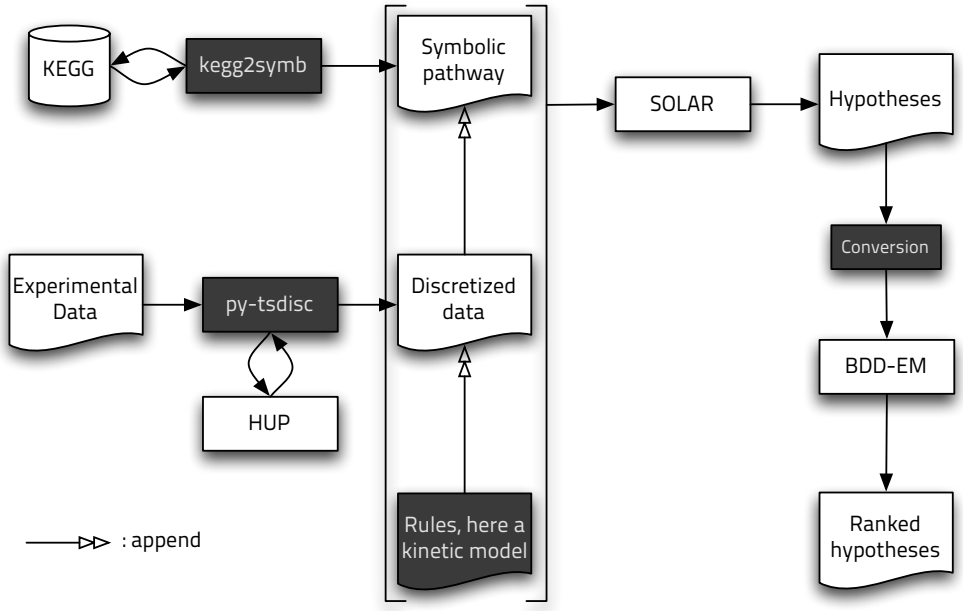


Figure 11: The **full process of hypothesis finding** (without enhancing the knowledge base). We want to discover new hypotheses (in the present work: abductibles) from experimental results. For that, we need to build a symbolic model of the studied pathway, we can use KEGG to extract such a model with `kegg2symb`. We have to discretize experimental data, we use HUP (HMM utility program) in conjunction with `py-tsdic` to generate a symbolic, discretized model of the data. We concatenate the model of the pathway, the data and the rules that will be used to infer new hypotheses in a big file that we give to SOLAR to generate hypotheses. Finally, we use BDD-EM to rank this hypotheses, so that we can study them in their order of interest. The result of the work of the current internship features a grey background.

level of concentration here. The assumption of a Markov process for the variations of concentrations of a metabolite can be justified by thinking that only the current level value is influencing the next value. This is not totally true, as we work with a complex system and for instance, with chained reactions of metabolites: $A \rightarrow B \rightarrow C \rightarrow A$, $C(A[t_0])$ influences $C(A[t_3])$. But this is a good assumption and our "pattern recognition" task of discretizing levels corresponds well to speech recognition [Rabiner 89] or part-of-speech tagging [Brants 00], both often done with HMMs. In mathematical terms, an HMM is made of a Markov chain: i.e. the current state $S(t)$ depends only on $S(t-1)$. It is an *hidden* Markov model because $S(t)$ cannot be observed. Instead, one can observe $X(t)$ which is the observation depending only on the value of $S(t)$, as in Fig.12.

4.2.2 Parameter Tying

We use continuous (Gaussian) HMMs (CHMMs) with **parameter tying**. CHMMs are HMMs working with continuous time series so they model the observable (output) values as mixture of Gaussians and have a lot of transitions (one for each new value) between the hidden states. Parameter tying is a solution to the problem of sharing the same symbolic levels in all the logic modeling in order to be able to assign the level of a compound to

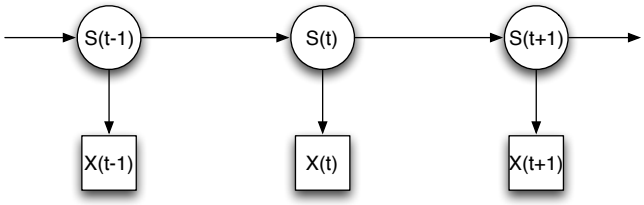


Figure 12: HMM: the conditional probability distribution of the hidden $S(t)$ only depends on the value of $S(t-1)$. The probability of an observed variable $X(t)$ only depends on its (time-) assiociated hidden variable $S(t)$.

another and be dealing with the same real values behind the scene. Basically, all the HMMs share a state space as well as the parameters in the output variables (i.e. means and variances), so that they produce discrete levels that are corresponding. Parameter tying is a notion often used in HMMs for speech recognition [Rabiner 89] and recently in statistical relational learning [De Raedt 08]. A typical example for parameter tying appears in a simple Markov chain, where the transition parameters are considered as time-invariant, i.e. $P(Z_{t+1} \mid Z_t) = P(Z_{t'+1} \mid Z_{t'})$ holds for any time indices t and t'. Here we say that the parameters $P(Z_{t+1} \mid Z_t)$ and $P(Z_{t'+1} \mid Z_{t'})$ ($t \neq t'$) are all tied. Note here that the sufficient statistics of tied parameters are shared accordingly. In our case, the mean and the variance for $X_t^{(n)}$, the output variable at time t in the HMM for the n-th metabolite ($n = 1, \ldots, N$), are tied with the mean and the variance for $X_{t'}^{(n')}$, respectively ($n \neq n'$ and $t \neq t'$). In contrast, to allow the concentration levels of N compounds to change in different ways, the transition parameters for $P(S_{t+1}^{(n)} \mid S_t^{(n)})$ are not tied with any for $P(S_{t'+1}^{(n')} \mid S_{t'}^{(n')})$, where $S_t^{(n)}$ is the hidden state variable at time t in the n-th HMM, $n \neq n'$ and $t \neq t'$.

4.2.3 Expectation-Maximization and Variational Bayes EM

An expectation-maximisation (EM) algorithm is used to find the maximum likelihood estimates of missing parameters in a probabilistic model by *iteratively repeating* the following *two steps*:

Given a likelihood function $L(\theta; S, X)$ with θ the parameters vector, S the unobserved data (missing values, here: the discrete levels) and X the observed data (here, the concentration values).

- Expectation step: compute the conditional expected value of the log likelihood function, with respect to the conditional distribution of X given S under θ.

$$Q(\theta|\theta^{(t)}) = E_{\theta^{(t)};S|X} \log L(\theta; S, X) \tag{4.1}$$

- Maximization step: find the parameters which maximizes it.

$$\theta^{(t+1)} = \arg\max_{\theta} Q(\theta|\theta^{(t)}) \tag{4.2}$$

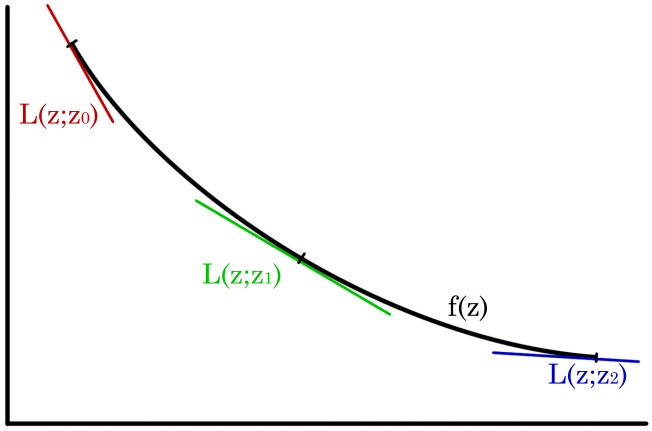


Figure 13: f is convex, any tangent $L(z; z_i)$ can be used as its lower bound

The variational Bayes EM algorithm is an implementation of the EM algorithm that uses the fact that the tangent of a convex at any point can always be used as its lower bound (see Fig.13) as an approximation:

$$\forall z : f(z) \geq L(z; z_0) = f(z_0) + \frac{\partial}{\partial z} f_{z=z_0}(z)(z - z_0) \tag{4.3}$$

The model parameters (θ) are treated as latent variables. For instance, the evidence is:

$$p(S|prior) = \int_\theta \int_X p(S, X, \theta|prior) dX d\theta \tag{4.4}$$

The joint posterior is contrained to be a factorized approximation:

$$q(X, \theta) \approx q_X(X) q_\theta(\theta) \tag{4.5}$$

(with the intention that q can be made very similar to the true posterior, this is measured by a dissimilarity function). The log-evidence is approximated to:

$$\begin{aligned}
\log p(S|prior) &= \log \int_\theta \int_X p(S, X, \theta|prior) dX d\theta && (4.6)\\
&= \log \int_\theta \int_X q(X, \theta) \frac{p(S, X, \theta|prior)}{q(X, \theta)} dX d\theta && (4.7)\\
&\geq \int_\theta \int_X q(X, \theta) \log \frac{p(S, X, \theta|prior)}{q(X, \theta)} dX d\theta && (4.8)\\
&\approx \int_\theta \int_X q_X(X) q_\theta(\theta) \log \frac{p(S, X, \theta|prior)}{q_X(X) q_\theta(\theta)} dX d\theta \text{ (with 4.5)} && (4.9)\\
\implies \log p(S|prior) &\geq F_{prior}(q_X(X), q_\theta(\theta)) && (4.10)
\end{aligned}$$

The VB-EM algorithm is as follows:

- VB expectation step: $q_X^{(t+1)}(X) = \arg\max_{q_X} F_{prior}(q_X(X), q_\theta^{(t)}(\theta))$
- VB maximization step: $q_\theta^{(t+1)}(\theta) = \arg\max_{q_\theta} F_{prior}(q_X^{(t+1)}(X), q_\theta(\theta))$

The purpose of the EM algorithm is to learn the parameters of the CHMMs that are the means and variances of the Gaussian corresponding to each level. For a 3-states ($\Leftrightarrow$ 3-discrete-levels) CHMM as in 14 (page 26), there are 3 Gaussians.

4.2.4 Discretization Process

The relevant discretized levels of concentration are computed through the EM algorithm with maximum a posteriori (MAP) estimation [Gauvain & Lee 94] (Baum-Welch algorithm) or through the variational Bayes EM (VB-EM) [Beal 03, Ji *et al.* 06]. We prefer this last method as it is shown [Beal 03] that variational free energy provides a more accurate approximation of the marginal log-likelihood than BIC or the Cheeseman-Stutz score. The discretization process itself simply consist in choosing with which discrete level will be mapped some numerical value of the time series. This is done by finding the most probable states sequence in the CHMM, once the parameters are all learned and tuned. For an example, see in appendix page 44. All of this is done with the help of HMM utility program (HUP) from Yoshitaka Kameya.

We first prepare N continuous HMMs (one for each metabolite), where each state variable takes a concentration level, and each output variable takes a measurement of concentration (2nd figure of Fig.14) and follows a univariate Gaussian distribution (1st figure of Fig.14). For knowing which number of states (levels) $k\star$ is better fitting for our set of time series (one for each metabolite), we test a range $[i \dots j]$. We then run VB-EM (or another EM algorithm) p times for each level and take the best outputs in regard to the scores (see below). We need to do numerous runs because variational Bayes is an approximate framework. We score the p CHMMs models (differing by their parameters, not the structure) of each level and take the best. We end up with $(j - i)$ CHMMs discretizing in $[i..j]$ levels, the "better fitting" number of levels has just been computed in an unsuppervised manner by taking the only one best score among this lasting $(j - i)$ CHMMs: this is the number of states. We can now recycle this very one CHMM and use it to find its most probable sequence (for instance with the Viterbi algorithm) that will be our discretized levels output sequence. **For example**, if we set $p = 100$ and test within the range $k \in [3 \dots 15]$, so we run 120 ($(15 - 3) \times 100$) times the VB-EM algorithm on the N continuous HMM with an increasing number of states (levels), from 3 to 15. At each "level-step", we keep only the highest scoring CHMM, we end up with 12 CHMMs. We take the best one, let it be the 3 states (3 levels) CHMM as for our simplified *Escherichia Coli* model and as in Fig.14. The discretization of the time series of concentrations is the most probable sequence of this CHMM: 1-1-2-3-3-3-2-1.

Then, we use a simple round-mean aggregation of them for time-sampling. We set a maximal number of time steps and look for the better fitting width and alignment for equal-width time intervals. We are currently developping a different discretization process (inside `py-tsdisc`) for time-series from molecular biology experiments that will discretize time and levels simultaneously but current results are already useable (see Table 1., Fig.18 and Fig.22) and that is what we based the results presented here on.

The **experimental** response observations of intracellular metabolites to a pulse of glucose were measured in continuous culture employing automatic stopped flow and manual fast sampling techniques in the time-span of seconds and milliseconds after the stimulus with glucose. The extracellular glucose, the intracellular metabolites: glucose6phosphate, fructose6phosphate, fructose1-6bisphosphate, glyceraldehyde3phosphate, phospho-enolpyruvate, pyruvate, 6phosphate-gluconate, glucose1phosphate as well as the cometabolites: atp, adp, amp, nad, nadh, nadp, nadph were measured using enzymatic methods or *High Performance Liquid Chromatography*. All the measured steady-state concentrations of the *E.Coli* experiment and their correspond-

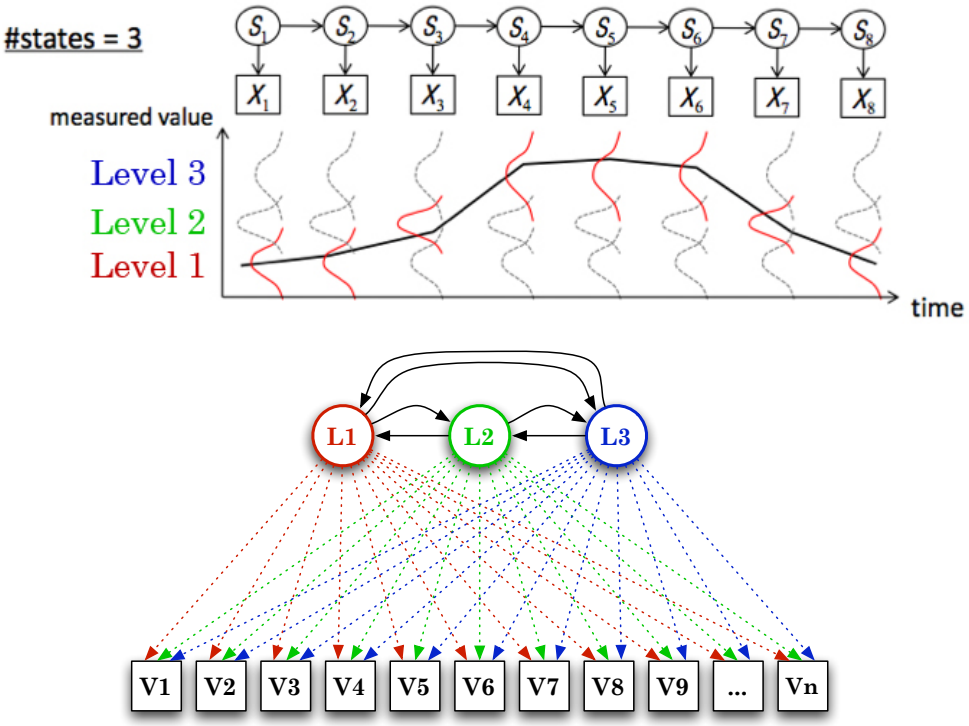


Figure 14: 3-states ($\leftrightarrow$ levels) continuous HMM discretizing one experimental time-series, where X_t is the measurement of concentration at time t taking a value in $[V_1 \dots V_n]$ and S_t is the hidden state that indicates the corresponding discretized level taking a value in $[L_1, L_2, L_3]$.

ing discrete levels are summarized in Table 1. We have obtained these discrete levels by the previously explained discretization method:

Metabolite	Concentration	Level	Metabolite	Concentration	Level
glucose	0.0556	0	g6p	3.480	2
f6p	0.600	0	fdp	0.272	0
gap	0.218	0	pep	2.670	2
pyr	2.670	2	6pg	0.808	1
g1p	0.653	0	amp	0.955	1
adp	0.595	0	atp	4.270	2
nadp	0.195	0	nadph	0.062	0
nad	1.470	1	nadh	0.100	0

Table 1. Concentrations (mM/L) and their discretized levels for steady states

4.3 Logic Model

4.3.1 Modeling of the Pathway

To obtain an understanding of the central metabolism, a logical model has been developed according to a kinetic model including the glycolysis and the pentose phosphate pathway for *Escherichia coli* [Chassagnole *et al.* 06]. The structure of such networks is commonly

displayed on metabolic maps (e.g. Fig.24 for the Glycolysis and Pentose Phosphate of *E.Coli*), where each reaction is described in terms of the participating enzyme, metabolites, cofactors and the reaction stoichiometry. These chemical reactions and transport steps can be thought of as the primary connections between metabolite pools that affect each other by mass action. Metabolites and enzymes can also interact through regulatory loops, *i.e.* feedback and feedforward interactions; these can be thought of as secondary connections for transmitting information through the network. Another important fea-

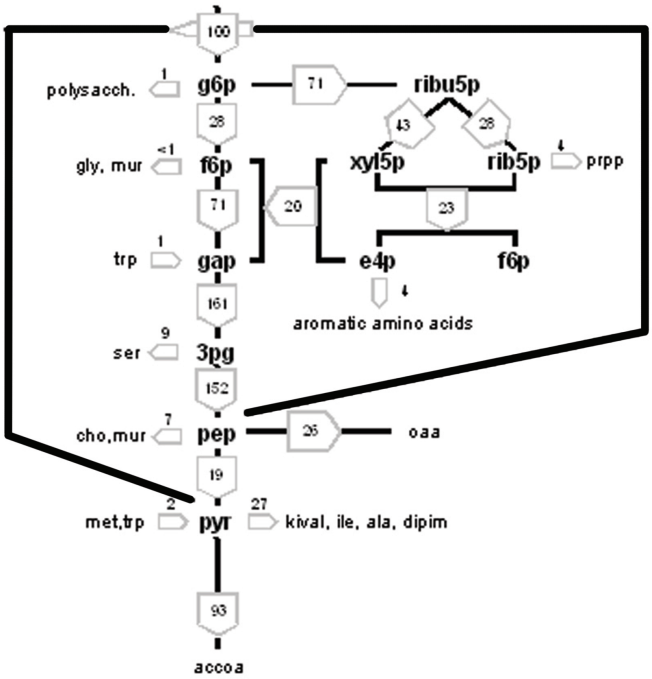


Figure 15: Simplified glycolysis and pentose phosphate pathways of *E.Coli*

ture of many metabolic systems is that they are divided into different compartments with the same metabolite sometimes occurring in two or more compartments. For the purpose of analysing the dynamic behaviour of such systems the compartmentalised pools of the same metabolite must be considered as separate metabolites even if they have the same chemical structure. The Fig.16 shows the simplified pathway that we modelized logically with relations `reaction(Substrate, Enzyme, Product).` For instance, the simplified glucolysis & pentose phosphate pathway for *Escherichia Coli* that was used with SOLAR can be found in Appendix page 44.

4.3.2 Abducing hypotheses with SOLAR

Inductive logic programming, used for induction or abduction, allows to deal with discrete levels (symbols) and qualitative rules [Doncescu *et al.* 07]. Given the background knowledge B and an observation E (example), the task of ILP is to find an hypothesis H such that:

$$B \wedge H \models E \tag{4.11}$$

$$\text{and } B \wedge H \quad is \quad consistent \tag{4.12}$$

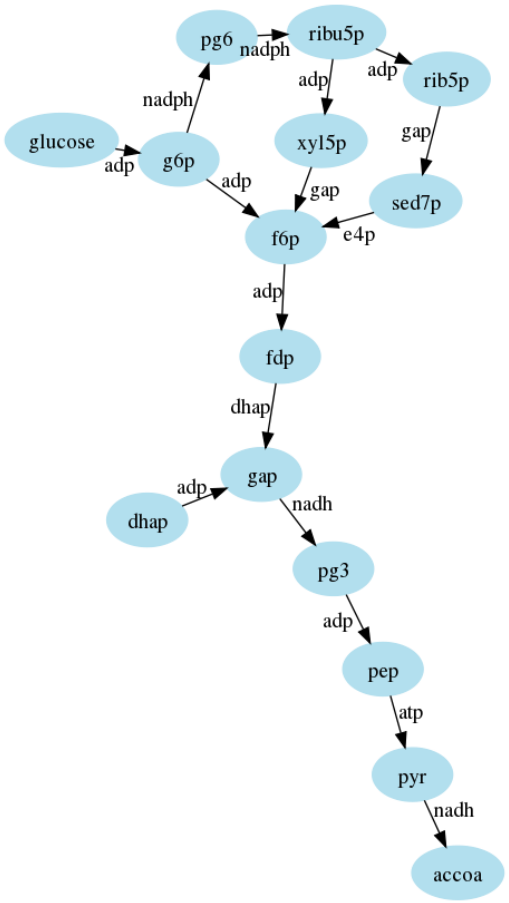


Figure 16: Simplified glycolysis and pentose phosphate pathways of *Escherichia Coli* automatically extracted from the symbolic knowledge base.

Inverse entailment [Muggleton 95, Inoue 04] enables us to compute H through deduction by using

$$B \wedge \neg E \models \neg H \tag{4.13}$$

$$\text{and } B \not\models \neg H \tag{4.14}$$

E and H being sets of literals, $\neg E$ and $\neg H$ are clauses. We are here interested in abducing what happens during the dynamical transition, but our approach stays valid for inducing general hypotheses as Inoue [Inoue 04] proposed a powerful method to handle inverse entailment (IE) for computing inductive hypotheses allowing for full clausal theories through consequence finding. $\neg H$ is constructed by a method called *CF-induction*, which computes *characteristic clauses* of $B \wedge \neg E$, selects its subset CC called the *bridge formula* from it, and generalizes $\neg CC$. CF-induction then realizes sound and complete hypotheses finding from *full clausal theories*, and not only definite clauses but also non-Horn clauses and integrity constraints can be constructed as H.

We used SOLAR [Nabeshima *et al.* 03] for generating the hypotheses, that are abducted in the current case. SOLAR is a competitive implementation of a consequence finding procedure based on SOL (Skipping Ordered Linear) tableaux which is *complete*

for finding minimal explanations. SOLAR is a part of CF-induction and can be used as an abductive procedure to infer an hypothesis H in the form of a set of literals. It can also output all the proofs (iterations) for each hypothesis of the returned set of hypotheses. For abducing hypotheses through IE, we need to input $B \wedge \neg E$ to deduce $\neg H$. Thus, all our observations $[e_1 \wedge \cdots \wedge e_n]$ will be inputed in SOLAR as a "top clause" $[\neg e_1, \ldots, \neg e_n]$ used together with B to deduce $\neg H$, the output of SOLAR . As the output of SOLAR is a conjunction of disjunctions, we have to negate it to have a disjunction of conjunctions: each conjunction being an hypothesis $H \in \mathcal{H}$.

4.3.3 Kinetic Modeling

Our kinetic logical model is based on the simplified Michaelis-Menten equation **(3.12)** which has here been represented by 3 background clauses using the `concentration(Compound, Level, Time)` predicate. This work was first suggested by Andrei Doncescu during a meeting. If we make the approximations for extreme values in:

$$[P]_{T+1} = V_m \frac{[S]_T}{[S]_T + K_m} + [P]_T \quad (3.12)$$

With only 3 levels, as we have in our discretization of *E.Coli* experiments, we will get the following simple rules:

$$\underline{[S] \ll K_m} \Rightarrow \frac{\Delta[P]}{\Delta T} = \frac{V_m}{K_M} \Rightarrow [P]_{T+1} = [P]_T$$

`reaction(S, P, Km)` $\wedge$ `concentration(S, 0, 0)` $\wedge$ `concentration (Km, 2, 0)` $\wedge$ `concentration(P, L, 0)` $\rightarrow$ `concentration(P, L, 1)` The concentration of the Product won't change between T and T+1 as the reaction will be *very* slow.

$$\underline{[S] \simeq K_m} \Rightarrow \frac{\Delta[P]}{\Delta T} = \frac{V_m}{2} \Rightarrow [P]_{T+1} = V_m/2 + [P]_T$$

`reaction(S, P, Km)` $\wedge$ `concentration(S, 1, 0)` $\wedge$ `concentration(Km, 1, 0)` $\wedge$ `concentration(P, L, 0)` $\rightarrow$ `concentration(P, L, 1)`

The concentration change of the Product between T and T+1 isn't big enough to switch from one level to another. This is an approximation and a consequence of our discretization.

$$\underline{[S] \gg K_m} \Rightarrow \frac{\Delta[P]}{\Delta T} = V_m \Rightarrow [P]_{T+1} = V_m + [P]_T$$

`reaction(S, P, Km)` $\wedge$ `concentration(S, 2, 0)` $\wedge$ `concentration(Km, 0, 0)` $\rightarrow$ `concentration(P, 2, 1)`

The reaction will be very quick and result in transforming all the Substrate into Product in one time step.

If we had more than 3 levels, we will either need more rules or a general procedure for handling our kinetic model. This is a current research topic being explored through the use of a "`compute(levels[], result)`" (with `levels[]` beeing a list) predicate implemented in the Java part of SOLAR . Furthermore, we made some simplifications in the pathways (see Fig.16) to be able to use only Michaelis-Menten kinetics, another

research topic is to extend our modeling to other type of reactions. These rules are subject to changes for others purposes.

We also added constraints about the unicity of levels at a given time to reduce the number of hypotheses while keeping consistency:

- `¬concentration(S, 0, T) ∨ ¬concentration(S, 1, T)`
- `¬concentration(S, 0, T) ∨ ¬concentration(S, 2, T)`
- `¬concentration(S, 1, T) ∨ ¬concentration(S, 2, T)`

4.4 Results

We chose to study the conjunction of glycolysis and pentose phosphate pathways for *E.Coli*. We simplified the pathway's model: we kept 16 relevant reactions (see Fig.16) and discretized the 16 experimental values (see Table 1). We added the 3 Michaelis-Menten based rules and the 3 constraints of unicity for the levels[3] . We had 15 abducibles corresponding to the unknown concentrations of chemical compounds before the transition to steady state. SOLAR , used for abduction, outputs 410 hypotheses that cover all this abducibles. With such a number, picking the right hypotheses should be done in an automated way. We also discuss here how we could improve the knowledge with the "most interesting" hypotheses.

4.4.1 Ranking the Hypotheses with BDD-EM

For that, Ishihata *et al.* [Ishihata *et al.* 08] proposed the BDD-EM algorithm that is an implementation of the expectation-maximization algorithm working on binary decision diagrams (BDDs), allowing it to deal with boolean functions. Inoue *et al.* [Inoue *et al.* 09] has applied the BDD-EM algorithm to rank hypotheses obtained through abduction. A BDD is a way to compress ("factorize") a boolean formula into an efficient datastructure (see Fig.17).

X_1	X_2	X_3	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

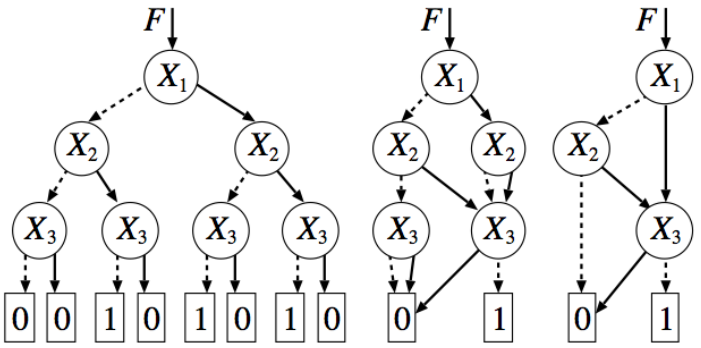


Figure 17: from left to right: the truth table of $[(X_1 \vee X_2) \wedge \neg X_3]$; its corresponding binary decision tree ; one corresponding ordered BDD ; one corresponding "factored" BDD: reduced ordered BDD (ROBDD).

To rank our $H_1, \ldots, H_n$ hypotheses by probability, we consider the finite set of ground atoms $\mathcal{A}$ that contains all the values that can take our `concentration (Compound,`

[3]The full model is avaible on the internet: `http://snippyhollow.free.fr/biomodels/`

`Level, Time)` and `reaction(Substrate, Product, Km)`. Each of the elements of $\mathcal{A}$ is a boolean variable. One of its subsets is the subset of abducibles Γ composed of all the possible values of `concentration(Compounds, Level, 0)`. With $\{A_i \in \mathcal{A} \mid \theta_i = P(A_i)\}$, we have to maximize the probability of the disjunction of hypotheses helped with the background knowledge B: $F = (H_1 \vee \cdots \vee H_n) \wedge B$ to set the good θ parameters (by the BDD-EM algorithm). F can still be to big to be retained as a BDD, so an optimisation F' of its size is obtained through the use of the minimal proofs for B and each H_i (see [Inoue *et al.* 09] for more details). Then, the BDD-EM algorithm computes the probabilities of ground atoms in $\mathcal{A}$ that maximizes the probability of F'. Finally, the probabilities of each hypotheses used for the ranking are computed as the products of the probabilities of literals appearing in each H_i.

We get 410 hypotheses in the form:

`H130 = concentration(glucose,0,1)∧concentration(g6p,2,1)`
`∧concentration(f6p,0,1)∧concentration(fdp,0,1)∧concentration(gap,0,1)`
`∧concentration(pg3,2,0)∧concentration(adp,0,0)∧concentration(pyr,2,1)`
`∧concentration(pg6,1,1)∧concentration(g1p,0,1)∧concentration(amp,1,1)`
`∧concentration(adp,0,1)∧concentration(atp,2,1)∧concentration(nadp,0,1)`
`∧concentration(nadph,0,1)∧concentration(nad,1,1)∧concentration(nadh,0,1)`

To sort them, we ran the EM algorithm on the BDDs corresponding to our hypotheses 10,000 times with random initializations. Note that if the comparison of this probabilities with each other is relevant, they shouldn't be taken as absolute probabilities. The 10 most probable abduced hypotheses and H378, that discovers all abductibles, are the following:

Hypothesis number	Probability	Abducted concentrations levels at T=0
H130	≈ 1.0	pg3: 2, adp: 0
H392	$4.879.E^{-1}$	sed7p: 0, e4p: 2, f6p: 0, pg3: 2, adp: 0
H216	$7.567.E^{-2}$	pg3: 2, adp: 0, pep: 0, atp: 2, pyr: 2
H196	$6.930.E^{-2}$	fdp: 0, dhap: 2, gap: 0, pg3: 2, adp: 0
H356	$5.621.E^{-2}$	pg3: 2, adp: 0, g6p: 1, nadph: 1
H94	$3.692.E^{-2}$	sed7p: 0, e4p: 2, f6p: 0, pg3: 2, adp: 0, pep: 0, atp: 2, pyr: 2
H251	$3.497.E^{-2}$	glucose: 2, adp: 0, pg3: 2
H286	$3.382.E^{-2}$	sed7p: 0, e4p: 2, f6p: 0, fdp: 0, dhap: 2, gap: 0, pg3: 2, adp: 0
H405	$2.796.E^{-2}$	pg3: 2, adp: 0, pep: 2, atp: 0
H167	$2.743.E^{-2}$	sed7p: 0, e4p: 2, f6p: 0, pg3: 2, adp: 0, g6p: 1, nadph: 1
.		
.		
H378	$1.974.E^{-8}$	glucose: 2, adp: 0, sed7p: 0, e4p: 2, f6p: 0, fdp: 0, dhap: 2, gap: 0, pg3: 2, pep: 0, atp: 2, pyr: 2, g6p: 0, nadph: 2, pg6: 1

Table 2. 10 most probable hypotheses and H378

4.4.2 Choosing the Hypotheses

It is needed to pick hypotheses that are consistent with the background knowledge and with each others. For example, if we apply a greedy algorithm (see below: **Algorithm 1**)

that picks hypothesis in decreasing probability order such that the hypothesis add some knowledge and that our enhanced knowledge is still consistent to this 10 first hypotheses, we will pick H130, H392, H216, H196, H356 and H251. It will result in discovering the concentrations at time T=0 of pg3: 2, adp: 0, sed7p: 0, e4p: 2, f6p: 0, pep: 0, atp: 2, pyr: 2, fdp: 0, gap: 0, dhap: 0, g6p: 1, nadph: 1, glucose: 2. We could go further and apply it on all the hypotheses for finding values for all the abducibles.

At first, we consider the background knowledge combined with the observations as our knowledge base. The goal of such an algorithm is to enhance (update) our knowledge base with discovered abductibles.

Algorithm 1 An algorithm to enhance the knowledge base: most probables firsts

```
knowledge ← knowledge_base
sorted_hypotheses ← sort(hypotheses)
while length(abductibles) > 0 && length(sorted_hypotheses) > 0 do
  tmp ← sorted_hypotheses.pop()
  if contains(tmp, abductibles) && consistent(tmp, knowledge) then
    knowledge.enhance(tmp)
    abductibles.remove(tmp)
  end if
end while
```

With the explicit functions *length*, *pop* (destructive), and:

- *sort* is a function that sorts the hypotheses by decreasing probability.
- *contains* is a function that returns statements of first argument contained in the second.
- *consistent* performs consistency checking of two theories and return True if they are consistent.
- *enhance* adds statements that are not yet present in the considered ("self", "this") knowledge.
- *remove* deletes statements from argument present in the considered ("self", "this") object (could make use of *contains*).

4.4.3 Interpretation

With an already existing simulator from Andrei Doncescu, we simulated the experiment and put approximations of this values (means of the level's "intervals") and the only value that did not seem to correspond was the one of nadph. Furthermore, this hypotheses are corresponding to our biological knowledge that pyruvate is a bottleneck [Peters-Wendisch *et al.* 01] and that the glucose that is totally consumed (see the left plot of Fig.18 from simulation) was in high concentration at the beginning of the experiment (pulse). Also, for other metabolites, as fructose6phosphate, the levels found through abduction are corresponding to the output of the simulation (see the right plot of Fig.18) with the same low level (0) before and after the dynamic transition.

The justification for an algorithm that enhances the background knowedge while keeping the consistancy (such as **Algorithm 1**) comes from the fact that the most probable

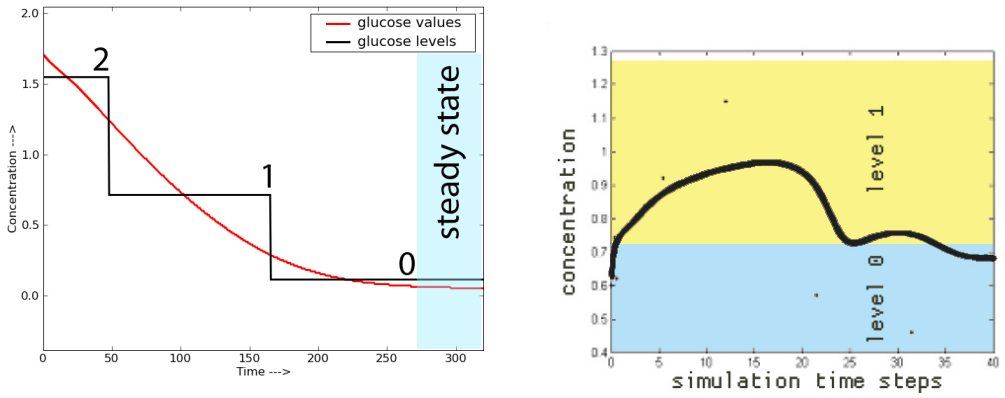


Figure 18: Left: Discretization in 3 levels of the concentration of glucose in the Glycolysis Pathway of *E.Coli* after an initial pulse. Right: Simulated evolution of fructose6phosphate during the whole experiment of pulse of glucose on *E.Coli*.

hypotheses doesn't infer a lot of values at once as shown in Fig.19, in which the red bars are showing $\#\{discovered\ abducibles\}$ / $\#\{abducibles\ (15)\}$. This results in a degree of liberty to pick the best combination of abducibles. While H378 discovers the 15 abductibles, it has a probability of only $1.974.E^{-8}$ (with regard to E^{-2} for the first ones). Therefore, we need either to complete the model (to have more restrictions, less hypotheses) or to choose what to do with this consequent degree of liberty with such an algorithm. While noticing that H378 is ranked very low, we could have chosen to pick from hypotheses that discover more abducibles by penalizing the fact of taking from too many different hypotheses with a formula such as the BIC [Schwarz 78]: $score = n\ln(\frac{error}{n}) + k\ln(n)$ with n being the number of chosen hypotheses, k the number of abducibles and $error$ the product of the probabilities of chosen hypotheses. The goal of such an algorithm would be to discover all abducibles (as **Algorithm 1**) while minimizing this score.

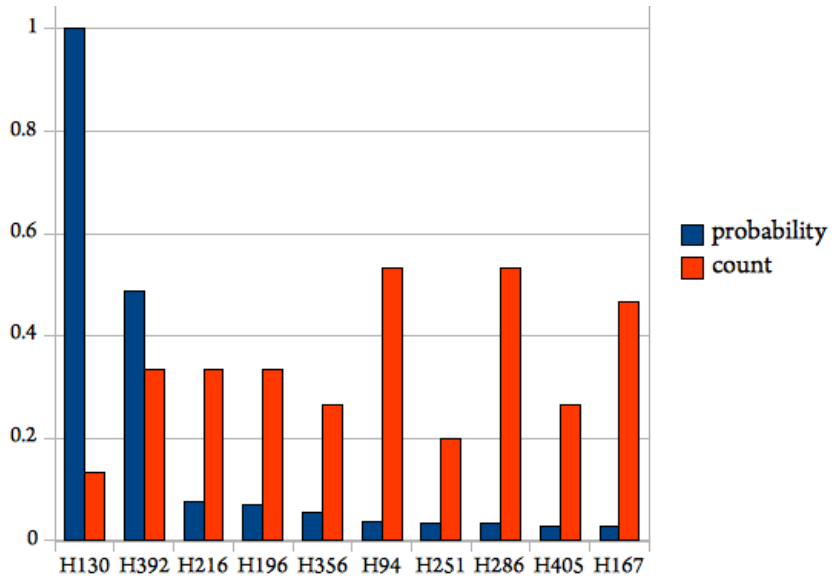


Figure 19: 10 first ranked hypotheses and their **normalized** coverage of abductibles.

5 Possible Extensions of this Work

5.1 Kinetic Modeling

Why is it important to have many levels ? To be able to handle the different K_m (Michaelis Menten constant): $K_m(\mathrm{ATP}) = 0.4\ mmol/L \longleftrightarrow K_m(\mathrm{HCO_3^-}) = 26\ mmol/L$.
With M levels, for example: `1 2 3 4 5 6 7`. **When** the need for $N < M$ levels arises, we do a projection, for example:

small: {`1, 2`} ; medium: {`3`} ; big: {`4, 5, 6, 7`}

Here, if we have only the 3 previously presented rules (Michaelis-Menten specific approximations).
How ? We center `medium` on the discrete level of the relevant K_m by doing `medium(A) :- level(A)==level(Km)` *in the context of a reaction* with a substract of product `A` and constant `Km`. This means that the discrete level that will correspond to `medium` for the current application of the three 3-levels-based rules will be the level taken by K_m. We made a **projection**.

Another way to deal with numerous levels is to change our modeling to allow for one (or many) more general(s) rule than the 3 rules showed above and resulting from approximations. This is what we try to achieve by introducing a new built-in predicate `compute` in SOLAR . As we take V_m and K_m from the literature, we can write it:
`reaction(X, E, Y, Km, Vm), concentration(X, LX, T), concentration(Y, LY, T)` ⇒ `compute(LX, LY, Vm, Km, LXX, LYY), concentration(X, LXX, T+1) concentration(Y, LYY, T+1)`
And in the other direction (from T to $T-1$):
`reaction(X, E, Y, Km, Vm), concentration(X, LXX, T), concentration(Y, LYY, T)` ⇒ `compute(LX, LY, Vm, Km, LXX, LYY), concentration(X, LX, T-1) concentration(Y, LY, T-1)`

`compute` would be executing the mathematical computations in the Java part of SOLAR while using a mathematical expression of the kinetics. For instance, we tried with the simplified Michaelis-Menten equation:

$$[P]_{T+1} = V_m \frac{[S]_T}{[S]_T + K_m} + [P]_T \quad (3.12)$$

Why shouldn't we directly deal with numerical values (without discretization) with `compute` ? It would be acceptable for abduction, but we would be venturing on the fields of analytical models with the wrong tools ; and how would we be so sure about our exact values ? It would *not* be acceptable for induction, as long as we want to have quite general inducted hypotheses. As explained before (section 3.1), we want to have a finer model but to still be able to generate general hypotheses with the existing induction methods. As levels account for some sort of "intervals", **why not** use an interval constraint logic programming approach as in [Benhamou 94]? This could be a future area of research. There could be some benefit of being actually aware of the values "behind" the levels symbols during the inductive process, but it would be complicating the hypothesis finding whereas the current approach still has to be studied more in depth (and could eventually give good results), while enjoying the benefits of having the real meanings of the levels

symbols dealt with *outside* the logic part.

Hidetomo Nabeshima provided consistent improvements in the efficiency of SOLAR 2.0 (not published yet) through pruning methods on top of SOL calculus [Nabeshima *et al.* 08] (based on tableaux calculus). SOLAR 2.0 also brought the possiblity to add special predicates (by extending the Op(erator)Checker class) that can do Java computations before returning their symbolic values in the tableau. This allowed us to write a prototype of a `compute` predicate but some problems arose. First, one has to deal only with integer: we solved that by having numerical values multiplied by a thousand (it could be more, we are on the very beginning of this research) and working with integers (and Euclidean division!). Secondly, the Java computation is executed for every node in the tableau: so, it will run this computation a lot of times with the same input symbols (arguments to the function "compute"). This will add a big overhead. We could solve this by precalculating all the values that could take this compute predicate and store it in a look-up table: `compute` takes 6 arguments that are discrete levels. For instance for the experiments on *Saccharomyces Cerevisiae* , we have determined that the better fitting number of levels is 9. For n arguments and k levels, we have k^n possible results for `compute`. So, for $n = 6$ and $k = 9$, `compute` will only take $9^6 = 531441$ different values. We can store them in a look-up table beforehand or the first (and last) time that we have to do a specific call to `compute` (if this argument combination is not found in the look-up table).

5.2 Building other Models

More generally, a predicate such as `compute` could be used for other models than a kinetic one. For instance, we can think about a structural modeling where one would have indentified the invariants of a metabolic pathway, as in [Reder 88]. For instance, Takehide Soh works on a SAT-solving approach for hypothesis finding based on symbolic pathways extracted from KEGG. Also, the multi-levels interactions with genes should be an interesting subject for logic modeling.

We have also been working on a model from a "pulse of glucose experiment" with *Saccharomyces Cerevisiae* . There are many possible continuations of this work to achieve the full hypothesis finding process. One could:

- simplify the pathway to allow for Michaelis-Menten kinetics only.
- add rules to allow for Hills and other allosteric kinetics (with more than 1substrate-1enzyme-1product reactions).
- use an implementation of `compute`.

We have used a notation of multiple substrates/enzymes/products reactions using a `complex` predicate in order to keep the syntaxical form of reaction. For instance, $S_1 + S_2 \rightarrow E_1S_1 + E_2S_2 \rightarrow E_1 + E_2 + P$ should be noted: `reaction(complex(S1, S2), complex(E1, E2), P)`. To write rules for Michaelis-Menten kinetics using the same approximations as 4.3.3 page 29, one could either expand them to more levels: but it will soon be too many rules (it is okay for 5 levels, not 9 as in *Saccharomyces Cerevisiae* 's experiment) **or** write a predicate for answering $\ll (A, B)$. This predicate should take into account the means and variances corresponding to each levels. Perhaps that $\ll (4, 5)$ is *True* but $\ll (1, 2)$ is *False*.

5.3 Hypothesis Finding

The preparation of data, the modeling and the inductive generation of hypotheses are very important, but we should not forget the other important part that aims at finding relevant hypotheses and using them: for instance, sorting them with BDD-EM [Ishihata *et al.* 08, Inoue *et al.* 09] seems very promising. There have been some works in this direction within the Japanese-French Symposium on Systems Biology. Petr Buryan worked on grammar-based genetic algorithms for scanning the hypothesis space. Gauvain Bourgne worked on hypothesis generation with a collective process [Bourgne & Inoue 09].

Algorithm 2 An algorithm to enhance the knowledge base: smallest number of hypotheses additions (hypotheses with the biggest abducibles coverage firsts)

```
knowledge ← knowledge_base
while length(abductibles) > 0 && length(sorted_hypotheses) > 0 do
  tmp ← hypotheses.covering_most(abductibles)
  if consistent(tmp, knowledge) then
    knowledge.enhance(tmp)
    abductibles.remove(tmp)
  end if
end while
```

With the functions functions $length$, $contains$, $consistent$, $enhance$, $remove$ as in **Algorithm 1** and:

- $covering\_most$ is a function that outputs the hypothesis covering the largest number of elements of the given (set) argument, here abducibles. It could delete it from the current container ("self", "this") for efficiency, but this is not necessary.

Nevertheless, it seems that having an efficient way to bootstrap the automatic knowledge discovery process is not yet achieved. It could (and should) be a very important future work for which we propose to enhance the knowledge base with an algorithm such as Algorithm 1 which takes the best hypotheses according to BDD-EM and add them to the knowledge base as long as it stays consistent and that we haven't explained all that we want to. The problems that arises now comes from the fact that we don't know beforehand which hypotheses we will discover by adding some other hypotheses to the knowledge base. The bruteforce exploration of such a combinatory[4] space has a very high computational cost. That's why there should be some research on how to make the knowledge base grow and which branches to explore. For that purpose, we also propose the Algorithm 2 which adds less hypotheses to "complete" the knowledge base: the hypotheses chosen for the enhancement are the one that cover the largest number of abductibles. It could still use BDD-EM to then take the most probable between two hypotheses that explain the same number of abducibles. Note that we could use the **number of covered observations as a ranking**, because this is exactly the purpose of inductive logic programming. This could be practically be done by using the proof of each hypothesis that comes with the proof in the output of SOLAR .

[4] if we have an initial knowledge base KB_0, then enhance it $KB_1 \leftarrow KB_0 \cup H_1$ **or** $KB_2 \leftarrow KB_0 \cup H_2$ **or** $KB_3 \leftarrow KB_0 \cup H_1 \cup H_2$. We then have a disjunction of 3 knowledge bases to explore and enhance differently for the next step, and so on...

6 Conclusion

We showed one way to **discretize biology experiments** into relevant levels to be used with ILP and logic programs in the large. An enhancement is already being investigated for the discretization. Moreover, based on this discretization of concentration into levels, we explained our processus to transform Michaelis-Menten analytical **kinetics equation into logic rules**, we are not aware of any previous work in this direction. We think that this approach improves the accuracy of the metabolic flux analysis. Allowing for other kinds of kinetic modeling (two substrate and/or two products reactions) would enable us to work with more complete models. As in [King *et al.* 05], this approach tries to enable the behaviour of many ordinary differential equations while considering a symbolic model.

The **more global approach** of discretizing experimental data and using it in conjunction with automatically generated symbolic pathways extracted from KEGG [Kanehisa & Goto 00, Kanehisa *et al.* 08] can be applied **regardless of the model chosen for infering new knowledge** (see Fig.11 page 22). This approach can be generically applied to turn quantitative results from systems biology into qualitative (symbolic) ones and should be seen as a step towards automation of hypothesis finding [King *et al.* 04]. The process of evaluating hypotheses thanks to BDD-EM [Inoue *et al.* 09] is seen as a good method to find relevant knowledge among the large quantity of processed data. Particularly here, we only need to *rank* (as opposed to *evaluate*) the hypotheses in regard to each others ("pairwise"). The practical validity of this full process has been shown by the results presented above while working in a well-known theoretical framework of inductive logic programming [Inoue 04, Mooney 97].

Still, our modeling can be improved. New highly probable knowledge could be appended to the knowledge base to try and discover new hypotheses, being it through abduction (to find missing observations) or full clausal theory induction (to find rules that govern the system). This idea of **revising the knowledge base** is already present in Ray *et al.* [Ray *et al.* 09] with a nonmonotonic approach. The pathway's symbolic model, time and concentration discretization could be finer. For instance, experiments dealing with more than 3 levels through a `compute` predicate implemented in SOLAR [Nabeshima *et al.* 03] and many time steps should be brought to a close on the Glycolysis and Pentose Phosphate pathways of another bacteria (representative of the eukaryote cells, this time), *Saccharomyces Cerevisiae* (yeast), with both real world data from experiments and simulated data. This will lead to use and test "iterative" (incremental) hypothesis finding process as this model is more complex and there are more discrete levels (9) and more discrete time steps.

This work can be seen as a proof of concept in the direction of mixing the real world experimental values, analytical methods and logic-based approaches. The current contribution can be found in Fig.11 page 22. We intend to modify (fork) the current hypothesis finding system found in Fig.4 page 13 towards something like Fig.20 page 38.

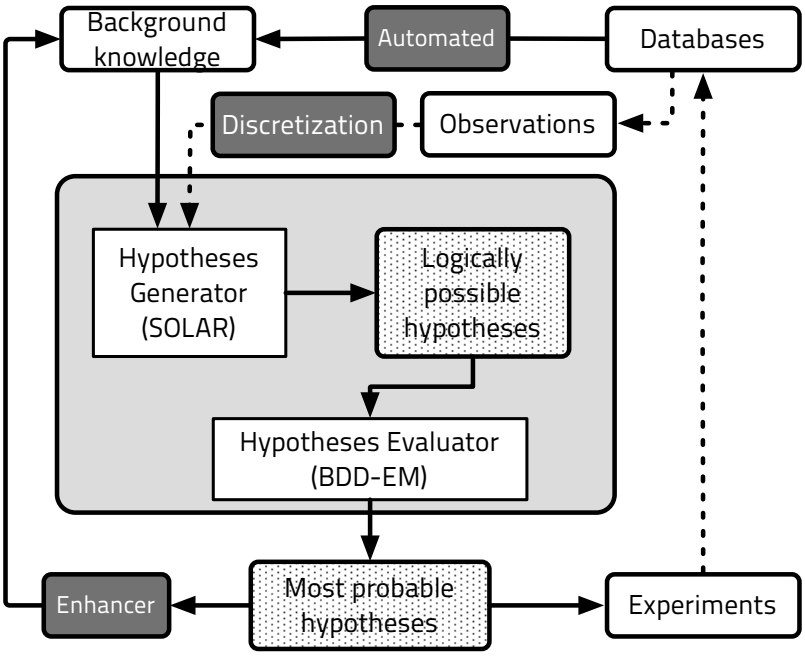


Figure 20: Enhanced biological hypothesis-finding system, the new parts have a grey (dark) background. To be compared with Fig.4 page 13.

Acknowledgments

This research is supported in part by the 2007-2009 JST-CNRS Strategic Japanese-French Cooperative Program. The author would like to thank Katsumi Inoue, Taisuke Sato, Andrei Doncescu, Pierre Bessière, Augustin Lux, Yoshitaka Kameya, Takehide Soh, Yoshitaka Yamamoto, Masakazu Ishihata, Hidetomo Nabeshima, Petr Buryan and Gauvain Bourgne for their all their advices, help, discussions, kindness. Thanks also to Nicolas Dumazet, Elsa Prieto, Benjamin Nègrevergne and all my friends in general for sharing pieces of smiles along the path of life. ☺

References

[Beal 03] Beal M. J.: Variational Algorithms for Approximate Bayesian Inference. PhD. Thesis, Gatsby Computational Neuroscience Unit, University College London (2003).

[Benhamou 94] Benhamou F.: Interval Constraint Logic Programming. Lecture Notes In Computer Science; Vol. 910 (1994).

[Bourgne & Inoue 09] Bourgne G., Inoue K.: Towards a model of collective knowledge discovery. In Pre-Proceedings of the Workshop on Abductive and Inductive Knowledge Developpement (July 2009).

[Brants 00] Brants T.: TnT - A Statistical Part-of-Speech Tagger. In Proceedings of the Sixth Applied Natural Language Processing Conference (2000).

[Chabrier-Rivier *et al.* 04] Chabrier-Rivier N., Fages F., Soliman S.: The biochemical abstract machine BIOCHAM. CMSB, Springer (2004).

[Chen *et al.* 07] Chen J., Muggleton S., Santos J.: Abductive stochastic logic programs for metabolic network inhibition learning. MLG (2007).

[Chassagnole *et al.* 06] Chassagnole C., Rodrigues J. C., Doncescu A. and Tianruo Yang L.: Differential evolutionary algorithms for in vivo dynamic analysis of glycolysis and pentose phosphate pathway in Escherichia Coli. In: A. Zomaya (ed.), *Parallel Computing in Bioinformatics and Computational Biology* (2006).

[Cheeseman & Stutz 95] Cheeseman P. and Stutz J.: Bayesian classification (AutoClass): Theory and results. Advances in Knowledge Discovery and Data Mining, pp.153-180, The MIT Press (1995).

[De Raedt 08] De Raedt L.: Logical and Relational Learning. Springer-Verlag (2008).

[Doncescu *et al.* 07] Doncescu, A., Inoue K., Yamamoto Y.: Knowledge Based Discovery in Systems Biology Using CF-Induction. LNCS No 4570, pages 395-404 (2007).

[Flach & Kakas 00] Flach P. A., Kakas A. C.: Abduction and induction: Essays on their relation and integration. Kluwer (2000).

[Franco & Canela 84] Franco R. and Canela E.: Computer simulation of purine metabolism. European Journal of Biochemistry, 144:305-315 (1984).

[Gauvain & Lee 94] Gauvain J.-L. and Lee C.-H.: Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains. IEEE Transactions on Speech and Audio Processing, Vol.2, Issue 2, pp.291-298 (1994).

[Geurts 01] Geurts P.: Pattern extraction for time-series classification. Proceedings of PKDD 2001, 5th European Conference on Principles of Data Mining and Knowledge Discovery, LNAI 2168, 115-127 (2001).

[Hofestädt & Thelen 98] Hofestädt R., Thelen S.: Quantitative Modeling of Biochemical Networks. In In Silico Biology (Journal), ISSN: 1386-6338, Vol.1, N 1/1998, p.39-53 (1998).

[Inoue 92] Inoue, K.: Linear resolution for consequence finding. Artificial Intelligence 56:301-353 (1992).

[Inoue 04] Inoue K.: Induction as consequence finding. Machine Learning, 55:109–135 (2004).

[Inoue *et al.* 09] Inoue K., Sato T., Ishihata M., Kameya Y. and Nabeshima H.: Evaluating abductive hypotheses using and EM algorithm on BDDs. IJCAI-09, to appear, in *Proceedings of IJCAI-09* (2009).

[Ishihata *et al.* 08] Ishihata M., Kameya Y., Sato T., Minato S.: Propositionalizing the EM algorithm by BDDs. Technical Report TR08-0004, Dept. of Computer Science, Tokyo Institute of Technology (2008)

[Ji *et al.* 06] Ji S., Krishnapuram B., and Carin L.: Variational Bayes for continuous hidden Markov models and its application to active learning. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.28, Issue 4, pp.522-532 (2006).

[Kanehisa & Goto 00] Kanehisa M. and Goto S.; KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Res. 28, 27-30 (2000).

[Kanehisa *et al.* 08] Kanehisa M., Araki M., Goto S., Hattori M., Hirakawa M., Itoh M., Katayama T., Kawashima S., Okuda S., Tokimatsu T., and Yamanishi Y.; KEGG for linking genomes to life and the environment. Nucleic Acids Res. 36, D480-D484 (2008).

[Keogh *et al.* 05] Keogh E., Lin J. and Fu A.: HOT SAX: efficiently finding the most unusual time series subsequence. 5th IEEE International Conference on Data Mining (2005).

[Kersting & De Raedt 01] Kersting K., De Raedt L.: Adaptive Bayesian Logic Programs. In: Rouveirol C., Sebag M. (eds.) ILP 2001. LNCS (LNAI), vol. 2157, Springer, Heidelberg (2001).

[King *et al.* 05] King R., Garrett S., Coghill G.: On the use of qualitative reasoning to simulate and identify metabolic pathways. Bioinformatics 21(9):2017-2026 (2005).

[King *et al.* 04] King R.D., Whelan K.E., Jones F.M., Reiser P.G.K., Bryant C.H., Muggleton S.H., Kell D.B. & Oliver S.G.: Functional genomic hypothesis generation and experimentation by a robot scientist. Nature 427, 247-252 (2004).

[Kitano 02] Kitano H.: Systems Biology Toward System-level Understanding of Biological Systems Kitano. In Science Vol. 295. no. 5560, pp. 1662-1664 (2002).

[Lee 67] Lee C.T.: A completeness theorem and computer program for finding theorems derivable from given axioms. Ph.D. thesis, Dept. of Electrical Engineering and Computer Science, University of California, Berkeley (1967).

[Lin *et al.* 07] Lin J., Keogh E., Wei L. and Lonardi S.: Experiencing SAX: a novel symbolic representation of time series. Data Mining and Knowledge Discovery, Vol.15, No.2, pp.107-144 (2007).

[Mauch *et al.* 00] Mauch K., Vaseghi S., Reuss M.: Quantitative Analysis of Metabolic and Signaling Pathways in Saaharomyces cerevisiae. Bioreaction Engineering: Modeling and Control, Springer Verlag (2000).

[Mirrahmi *et al.* 05] Mirrahimi E., Rouchon P., Turinici G.: Lyapunov control of bilinear Schrödinger equations. Automatica 41(11):1987-1994 (2005).

[Mooney 97] R. J. Mooney: Integrating abduction and induction in machine learning. In Working Notes of the IJCAI97 Workshop on Abduction and Induction in AI, 37–42 (1997).

[Mörchen *et al.* 05] Mörchen F., Ultsch A.: Optimizing time Series Discretization for Knowledge Discovery. Proc. of KDD-05, pp.660-665 (2005).

[Muggleton 95] Muggleton S.: Inverse Entailment and Progol. New Generation Computing, Special issue on Inductive Logic Programming, 13:3-4 pp245-286 (1995).

[Muggleton & Chen 08] Muggleton S., Chen J.: A Behavioral Comparison of Some Probabilistic Logic Models. Lecture Notes in Computer Science, Springer (2008).

[Nabeshima *et al.* 03] Nabeshima H., Iwanuma K., and Inoue K.: SOLAR : A Consequence Finding System for Advanced Reasoning. Proceedings of the 11th International Conference TABLEAUX 2003, LNAI, Vol. 2796, pp. 257-263, Springer (2003).

[Nabeshima *et al.* 08] Nabeshima H., Iwanuma K., and Inoue K.: Completeness of pruning methods for consequence finding procedure SOL. Proceedings of the LPAR 2008 Workshops, Knowledge Exchange: Automated Provers and Proof Assistants, and the 7th International Workshop on the Implementation of Logics (2008).

[Nienhuys-Cheng & De Wolf 97] Nienhuys-Cheng S.H., De Wolf R.: Foundations of inductive logic programming. LNAI 1228, Springer (1997).

[Peters-Wendisch *et al.* 01] Peters-Wendisch P. G., Schiel B., Wendisch V. F., Efstratios Katsoulidis E., Möckel B.: Pyruvate Carboxylase is a Major Bottleneck for Glutamate and Lysine Production by Corynebacterium glutamicum. J. Mol. Microbiol. Biotechnol. 3(2) (2001).

[Rabiner 89] Rabiner L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. Proc. of the IEEE Vol.77, No.2 (1989).

[Ray *et al.* 09] Ray O., Whelan K., King R.: A nonmonotonic logical approach for modelling and revising metabolic networks. Complex, Intelligent and Software Intensive Systems, IEEE Computer Society (2009).

[Reder 88] Reder C.: Metabolic control theory: a structural approach. Journal of theoretical Biology (1988).

[Sato 98] Sato T.: Modeling Scientific Theories as PRISM Programs. In Proceedings of ECAI98 Workshop on Machine Discovery (1998).

[Savitzky & Golay 64] Savitzky A., Golay M.J.E.: Smoothing and Differentiation of Data by Simplified Least Squares Procedures. Analytical Chemistry 36 (8): 1627–1639 (1964).

[Schwarz 78] Schwarz G.: Estimating the dimension of a model. Annals of Statistics, Vol.6, No.2, pp.461-464 (1978).

[Tamaddoni-Nezhad *et al.* 06] Tamaddoni-Nezhad A., Chaleil R., Kakas A., and Muggleton S.H.: Application of abductive ILP to learning metabolic network inhibition from temporal data. Machine Learning, 64:209-230 (2006).

[Yamamoto 97] Yamamoto A.: Which hypotheses can be found with inverse entailment? Lecture notes in computer science (LNCS) ISSUE 1297, pages 296-308, Springer (1997).

[Yamamoto *et al.* 08] Yamamoto Y., Inoue K., Doncescu A.: Estimation of Possible Reaction States in Metabolic Pathways using Inductive Logic Programming. In Proc. of 22nd International Conference on Advanced Information Networking and Applications (2008).

[Waser *et al.* 83] Waser M., Garfinkel L., Kohn C., and GarfinkelD .: Computer Modeling of Muscle Phosphofructokinase. Journal of Theoretical Biology, 103:295-312 (1983).

Glossary

abduction find hypotheses that are ground or existentially quantified formulaes (missing facts or data). 3, 7

allosteric regulation regulation of an enzyme (or other protein) by binding an effector module at a protein's site that is different from the active one. 10

anabolism synthesis of large molecules and structural components of the cell (proteins, RNA, DNA, lipids) from smaller substrates and chemical energy. 6

Baum-Welch algorithm generalized EM algorithm that computes maximum likelihood estimates and posterior mode estimates for the parameters (transition and emission probabilities) of an HMM, when given only emissions as training data.. 25

catabolism degradation of an external substrate (ex: Sugar) into smaller chemical products to provide energy (ex: stored in ATP). 6

clausal theory finite conjunctive set of clauses. 7

clause disjunction of litterals, $\bigvee_{i=1}^{n} A_i$. 3, 7

expectation-maximization iterative algorithm to find the maximum likelihood estimates of parameters in a probabilistic model. 23, 30

hidden Markov model ($\subset$ Bayesian networks) statistical model in which the system modeled is assumed to be a Markov process with unobserved states. 17, 21

Horn clause definite or negative clause, ex: $\{\neg A \vee \neg B \vee C\} \Leftrightarrow \{A \wedge B \Rightarrow C\}$. 7

induction find hypotheses that are unversally quantified formulaes (missing rules). 3, 7

inhibition decreasing an enzyme activity (and so reaction rate) by binding some molecule to it at a strategic location. 6

integrity constraint negative clause, ex: $\{\neg A \vee \neg B \vee \neg C\}$. 7

likelihood function given a parametrized density function $x \mapsto f(x|\theta)$, the likehood function is $\theta \mapsto f(x|\theta)$,written $L(\theta; x) = f(x|\theta)$. 23

metabolic pathway sequence of reactions occuring within the cell, which are interconnected via substrates, catalyzed by enzymes and often requiring other cofactors. 3, 4, 6

metabolism set of reactions that maintain life. 3

metabolite intermediate and/or final chemical product of a metabolism. 3, 5

subsumption (subsomption) $C\theta$-subsumes D iff $\exists\theta$ such that, syntactically, $C\theta \subseteq D$. We also say that C is more general than D.. 7

Appendix

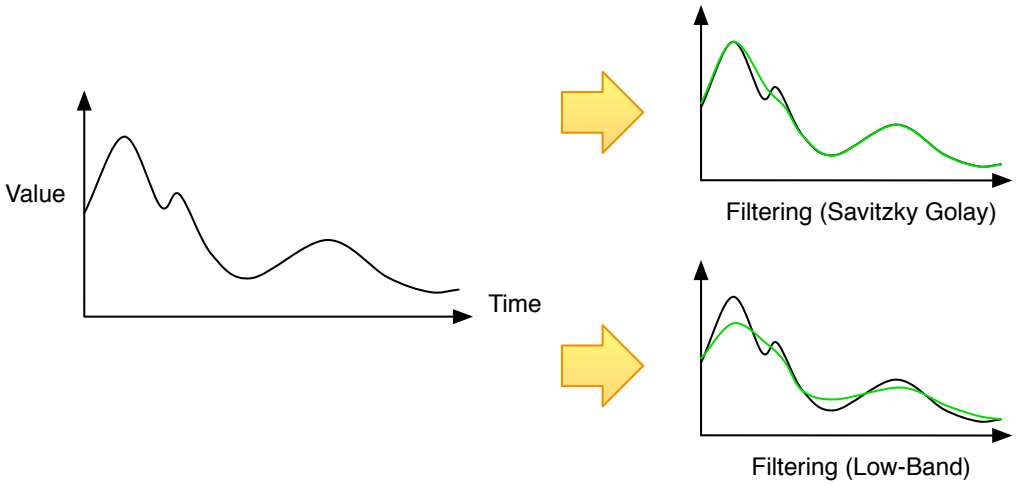


Figure 21: Savitzky-Golay filtering compared to low-pass filtering.

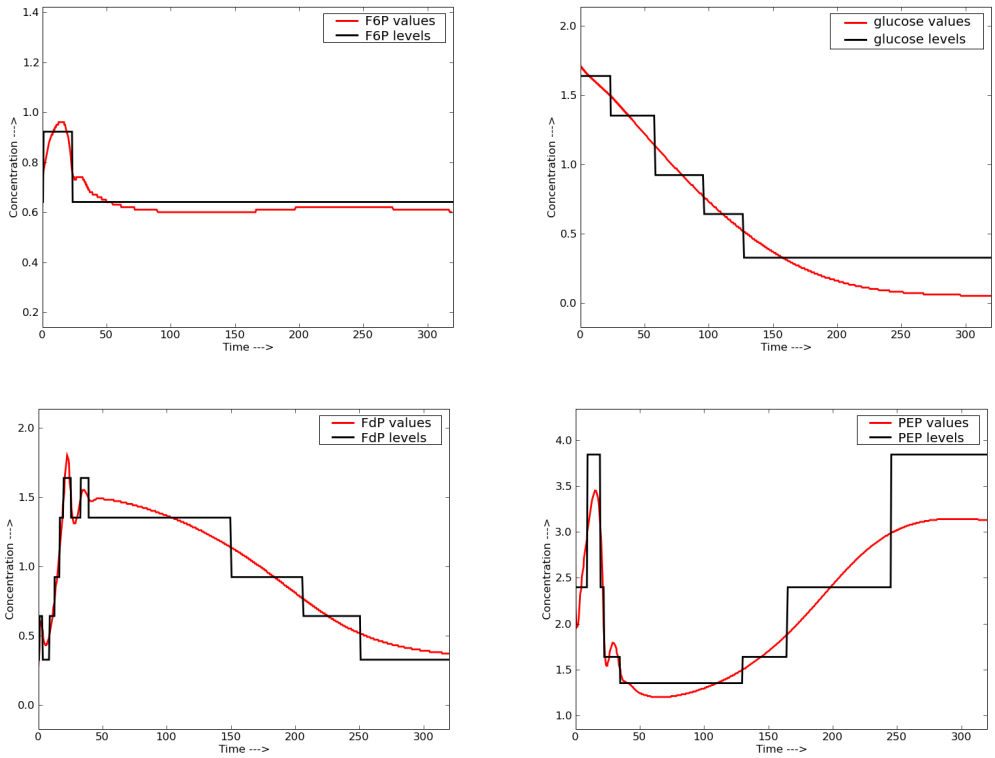


Figure 22: Discretization with 7 common levels of the concentration of 4 metabolites in the Glycolysis Pathway of *S.Cerevisiae* after an initial pulse of glucose.

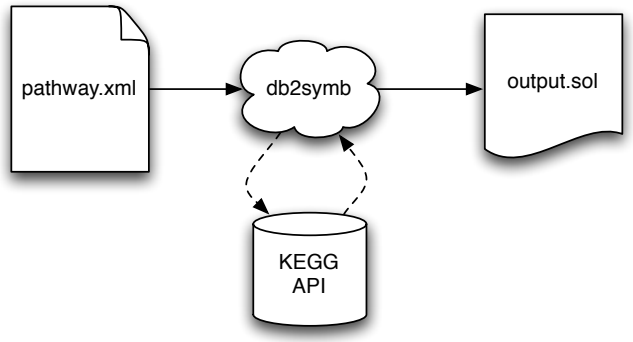


Figure 23: Overview of `kegg2symb` (old name, `db2symb`)

6.1 (Short) Example of the discretization of a time series

Input: time series of concentration of metabolites A and B:

time (in s.)	[A] (in mmol/L)	[B] (in mmol/L)
0.1	1.92	0.22
0.2	1.69	0.23
0.3	1.51	0.25
0.4	1.42	0.38
0.5	1.33	0.42
0.6	1.25	0.44

Intermediate output of the CHMM: most probable sequence of the HMM encoding the levels of A and B as hidden variables.

time (in s.)	level of A	level of B
0.1	4	1
0.2	4	1
0.3	4	1
0.4	4	2
0.5	3	2
0.6	3	2

Final output:

```
concentration(A, 4, 0)
concentration(A, 3, 1)
concentration(B, 1, 0)
concentration(B, 2, 1)
```

6.2 Glycolysis & Pentose Phosphate Pathway for *E.Coli*

The SOLAR input file is as follows:

```
%% Glycosys + Pentose Phosphate pathways
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Glycolysis: 9 enzyme reactions
%% Pentose-Phosphate: 7 enzyme reactions
%% 33 compounds
```

```
%% 18 enzymes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%
%%%% Reactions
%%%%%%%%%%%%%%%%

%%Predicates:
%reaction_1(Substrate, Enzyme1, Product).
%reaction_2(Substrate1,Enzyme1, Product,Enzyme2).
%reaction_3(Substrate1, Substrate2, Substrate3, Enzyme, Product).

% Phosphotransferase system
cnf(r1_1, axiom, [reaction_2(glucose,atp,g6p,adp)]).

%Embden-Meyerhof-Parnas pathway
cnf(r2_1, axiom, [reaction_2(g6p,atp,f6p,adp)]).
cnf(r2_2, axiom, [reaction_2(f6p,atp,fdp,adp)]).
cnf(r2_3, axiom, [reaction_2(fdp,atp,gap,dhap)]).
cnf(r2_4, axiom, [reaction_2(dhap,atp,gap,adp)]).
cnf(r2_5, axiom, [reaction_2(gap,nad,pg3,nadh)]).
cnf(r2_6, axiom, [reaction_2(pg3,atp,pep,adp)]).
cnf(r2_7, axiom, [reaction_2(pep,adp,pyr,atp)]).
cnf(r2_8, axiom, [reaction_2(pyr,nad,accoa,nadh)]).

%Pentose phosphate pathway
cnf(r3_1, axiom, [reaction_2(g6p,nadp,pg6,nadph)]).
cnf(r3_2, axiom, [reaction_2(pg6,nadp,ribu5p,nadph)]).
cnf(r3_3, axiom, [reaction_2(ribu5p,atp,rib5p,adp)]).
cnf(r3_4, axiom, [reaction_2(ribu5p,atp,xyl5p, adp)]).
cnf(r3_5, axiom, [reaction_2(rib5p,xyl5p,sed7p,gap)]).
cnf(r3_6, axiom, [reaction_2(sed7p,gap,f6p,e4p)]).
cnf(r3_7, axiom, [reaction_2(xyl5p,e4p,f6p,gap)]).

%%%%%%%%%%%%%%%%%%%%%%
%% Causal relations
%%%%%%%%%%%%%%%%%%%%%%

%%% On the predicate ``reaction_2''

cnf(un, axiom, [
       -reaction_2(Substrate1,Enzyme1, Product, Enzyme2),
       -concentration(Substrate1, 0, 0),
       -concentration(Enzyme2, 2, 0),
       -concentration(Product, L, 0),
       concentration(Product, L, 1)
       %%% concentration(Product, 0, 1)
]).

cnf(un, axiom, [
       -reaction_2(Substrate1, Enzyme1, Product, Enzyme2),
       -concentration(Substrate1, 1, 0),
       -concentration(Enzyme2, 1, 0),
       concentration(Product, 1, 1)
]).
```

```
cnf(un, axiom, [
       -reaction_2(Substrate1, Enzyme1, Product, Enzyme2),
       -concentration(Substrate1, 2, 0),
       -concentration(Enzyme2, 0, 0),
       concentration(Product, 2, 1)
]).

%%% On the fact that a given product can't have 2 different levels at T

cnf(un, axiom, [-concentration(S, 0, T), -concentration(S, 1, T)]).
cnf(un, axiom, [-concentration(S, 0, T), -concentration(S, 2, T)]).
cnf(un, axiom, [-concentration(S, 1, T), -concentration(S, 2, T)]).

%%%%%%%%%%%%%%%%%%%%%%
%% Observation(s)
%%%%%%%%%%%%%%%%%%%%%%

%%%Test examples
cnf(ex, top_clause, [-concentration(glucose,0,1),
 -concentration(g6p,2,1),
 -concentration(f6p,0,1),
 -concentration(fdp,0,1),
 -concentration(gap,0,1),
 -concentration(pep,2,1),
 -concentration(pyr,2,1),
 -concentration(pg6,1,1),
 -concentration(g1p,0,1),
 -concentration(amp,1,1),
 -concentration(adp,0,1),
 -concentration(atp,2,1),
 -concentration(nadp,0,1),
 -concentration(nadph,0,1),
 -concentration(nad,1,1),
 -concentration(nadh,0,1)]).

%%%%%%%%%%%%%%%%%%%%%%
%%% Inductive Bias
%%%%%%%%%%%%%%%%%%%%%%

production_field([+-concentration(_,_,_)]<=40).
```

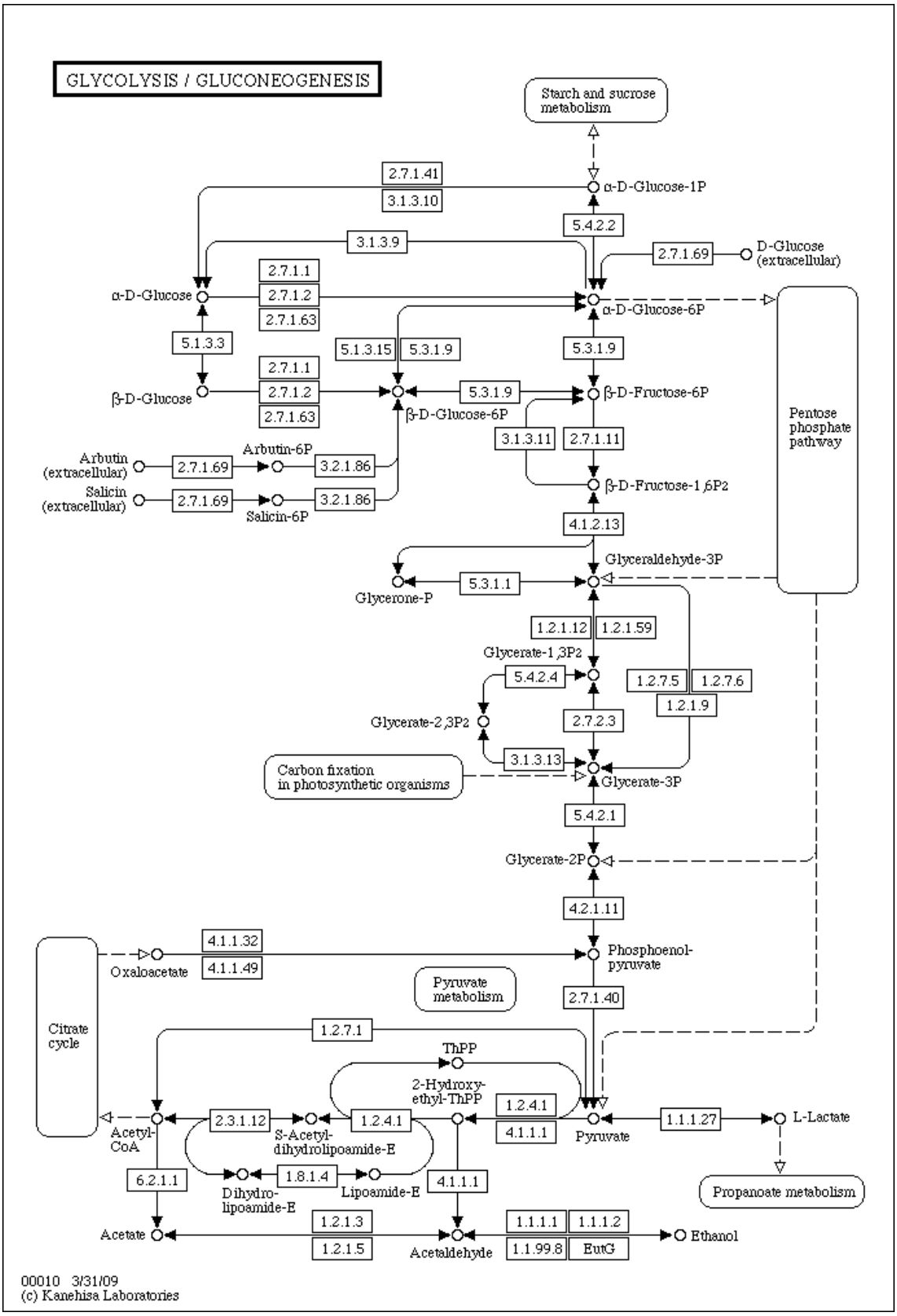


Figure 24: General glycolysis / glucogenesis pathway

6.3 ILP 2009 Poster

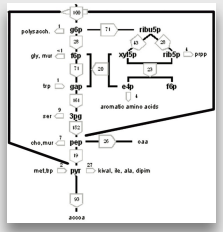
Kinetic Models for Logic Based Hypothesis Finding in Metabolic Pathways

Gabriel Synnaeve[1,3], Andrei Doncescu[2,3], Katsumi Inoue[3]

1. Grenoble University (Grenoble)
2. LAAS-CNRS (Toulouse)
3. National Institute of Informatics (Tokyo)

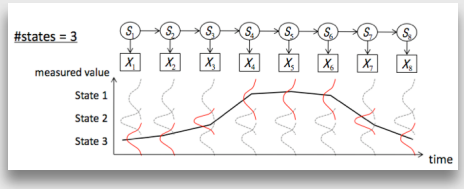
Metabolic Pathways

We propose a logical model of Glycolysis and Pentose phosphate pathways of *E.Coli* that enables us to apprehend better the dynamical response of a biological pathway to a pulse of glucose. This approach is based on a kinetic modeling of the evolution of the concentration of metabolites in a metabolic pathway. The majority of kinetic models in biology are studied with ordinary differential equations (ODE). We try here to understand the phenomenon described by a complex system, such as a cell, by doing a qualitative study with discrete levels of concentration of metabolites processed by inductive logic programming (ILP) rules.



Discretization

We first prepare N continuous HMMs (one for each metabolite), where each state variable takes a concentration level, and each output variable takes a concentration and follows a univariate Gaussian distribution. All the HMMs share a state space as well as the parameters in the output variables so that they produce discrete levels that are corresponding.



Michaelis-Menten Kinetics

Michaelis-Menten kinetic model is the most general representation for a non-linear allosteric regulation system. Assumptions made:

- quasi steady-state (d[ES]/dt ≈ 0)
- one substrate ↔ one product reactions

Reaction: $E + S \rightleftharpoons^{k_1}_{k_{-1}} ES \rightarrow^{k_2} E + P$

MM equation: $\frac{d[P]}{dt} = V_m \frac{[S]}{[S] + K_m}$

Simplifications: $\Longrightarrow [P]_{T+1} = V_m \frac{[S]_T}{[S]_T + K_m} + [P]_T$

Modeling

We use **Inverse entailment (IE) for abduction** with **SOLAR** (Nabeshima, Iwanuma & Inoue 2003)

- ***B***: background knowledge, full clausal theory
- ***E***: examples, conjunction of literals (¬E is a clause)
- ***H***: hypotheses, conjunctions of literals (¬H is a clause)

Computing a hypothesis H can be done deductively:

$$\boldsymbol{B} \wedge \neg \boldsymbol{E} \models \neg \boldsymbol{H}$$

We compose a background knowledge with reactions, levels corresponding to measured concentrations at the end of the experiment,and this rules (corresponding to the kinetic model):

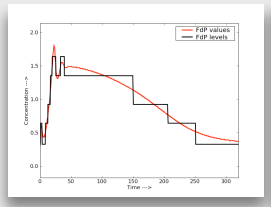
- $[S] \ll K_m \Rightarrow [P]_{T+1} = [P]_T$

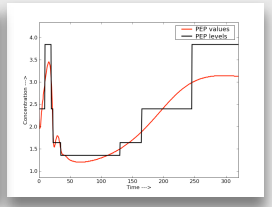
  Conc(s, 0, 0) ∧ Conc (Km, 2, 0) ∧ Conc(p, L, 0) → Conc(p, L, 1)
- $[S] \simeq K_m \Rightarrow [P]_{T+1} = V_m/2 + [P]_T$

  Conc(S, 1, 0) ∧ Conc (Km, 1, 0) ∧ Conc(p, L, 0) → Conc(P, L, 1)
- $[S] \gg K_m \Rightarrow Vm \Rightarrow [P]_{T+1} = V_m + [P]_T$

  Conc(S, 2, 0) ∧ Conc (Km, 0, 0) → Conc(P, 2, 1)

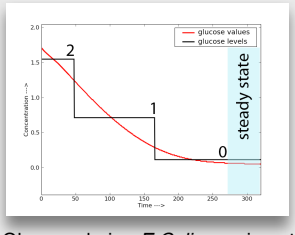
Results





Discretization of fructose-1,6-bisphosphate and phosphoenolpyruvate in another experiment (*S.Ce*).

The results obtained from SOLAR are corresponding to our biological knowledge that pyruvate is a bottleneck and that the glucose, that was in high concentration at the beginning, is totally consumed (see right).



Glucose during *E.Coli* experiment

Emails, respectively: gabriel.synnaeve@gmail.con, adoncesc@nii.ac.jp, ki@nii.ac.jp